TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

ILKKA VEIMA

FOOTBALL CLUB CONTENT MANAGEMENT SYSTEM

Master of Science thesis

Examiner: Prof. Kari Systä
Examiner and topic approved by the
Faculty Council of the Faculty of
Computing and Electrical Engineering
on 9th of September 2015

ABSTRACT

ILKKA VEIMA: Football club content management system
Tampere University of Technology
Master of Science thesis, 86 pages, 10 Appendix pages
August 2016
Master's Degree Programme in Electrical engineering
Major: Embedded systems
Examiner: Prof. Kari Systä
Keywords: CMS, sports club, web hosting

In this thesis a new web content management system is impelemented for a football club. The user, functional and system requirements for the implementation are documented.

Different alternatives for the impelemntation are compared to find the most suitable. Three approaches are considered: available services, content management systems and a custom implementation. A number of available services were found and a description of each is given. Pros and cons of each service is presented and features are compared to the requirements. Wordpress, Joompla and Drupal content management systems are looked at to see if they can be configured to meet the requirements. Ways to speed up the development of a custom solutions are presented. An available service to satisfy all the requirements was not found. A content management system could not be configured to meet the requirements either. A custom solution was chosen as the implementation approach.

Different hosting solutions were compared to find the cheapest and easy to maintain alternative. OpenShift was selected as it can be used for free, it is reliable and easy to use.

Pencilblue content management system became the basis of the custom implementation because it has features that made the development easier. Two plugins were build to extend the Pencilblue content management system. A plugin for managing the football club content and a theme. Description of the Pencilblue content management system and the two plugins are given. Results of a performance test are discribed and discussed. Finally the custom implementation is compared to the requirements and further development is discussed.

TIIVISTELMÄ

ILKKA VEIMA: Jalkapalloklubin sisällönhallintajärestelmä
Tampereen teknillinen yliopisto
Diplomityö, 86 sivua, 10 liitesivua
Elokuu 2016
Sähkötekniikan koulutusohjelma
Pääaine: Sulautetut järjestelmät
Tarkastajat: Prof. Kari Systä
Avainsanat: Sisällönhallintajärjestelmä, urheilukerho, verkkopalvelu

Tässä työssä toteutetaan uusi verkkosisällönhallintajärjestelmä jalkapallojoukkueelle. Aluksi kuvataan järjeslmän käyttäjä, tominnalliset ja yleiset vaatimukset.

Eri toteutusvaihtoehtoja vertaillaan parhaan toteutustavan löytämiseksi. Kolmea eri tapaa vertaillaan: Valmiin palvelun käyttö, sisällönhallintajärestelmän käyttö ja oma toteutus. Valmiita palveluita löytyi useita ja jokaisesta on tehty lyhyt kuvaus. Palveluiden hyvät ja huonot puolet esitetään ja niiden ominaisuuksia verrataan vaatimuksiin. Wordpress, Joomla ja Drupal sisällönhallintajärestelmiä tarkastellan, jotta saadaan selville voidaanko vaatimukset täyttää konfiguroimalla jokin niistä.

Valmiista palveluilsta ei löytynyt sellaista, joka olisi täyttänyt kaikki järjestelmälle asetetut vaatimukset. Myöskään sisällönhallintajärestelmän konfiguroinnilla ei saatu täytettyä kaikkia vaatimuksia. Oma toteutus valittiin toteutustavaksi.

Eri verkkopalveluita vertaillaan, jotta löydetään halvin ja helpoiten ylläpidettävä vaihtoehto järjestelmän ajamiselle. OpenShift verkkopalvelu valittiin, koska sitä voi käyttää ilmaiseksi, palvelu on luotettava ja sitä on helppo käyttää.

Pencilblue sisällönhallintajärestelmä valitiin toteutuksen pohjaksi, koska sen ominaisuudet helpottivat järjestelmän toteutusta. Kaksi lisäosaa toteutettiin sisällönhallintajärestelmän laajentamiseksi. Toinen lisäosa vastaa jalkapallojoukkueen sisällön hallinnasta ja toinen järjestelmän ulkoasusta. Työssä kuvataan sekä Pencilblue sisällönhallintajärestelmä, että kaksi toteutuettua lisäosaa. Suorituskykytestin tulokset esitetään ja eritellään. Lopuksi järjestelmän toteutusta verrataan järjestelmän vaatimuksiin ja jatkokehitys ideat käsitellään.

ACKNOWLEDGEMENTS

Ever since the founding of the Roxbury JK, I have wanted to build a great website for the team. However, lack of skill, lack of time and all other hobbies always came in the way. It was great that I could combine this ambition and the final phase of my university education.

The subject of this thesis may seem simple at first, but I learned great many things. The project was a dive into web technologies, but it also teached me the difficulty of combining work and a demanding projct, time management and other skills. After the ups and downs of the journey I feel I have grown more resilient and understanding. The non-technical skill may even prove to be the most useful lessons of this project.

I want to thank Kari Systä for the flexible guidance throughout the project. Thanks to Kari and Tuukka Ruotsala for the input and feedback. I thank everyone in Roxbury JK for the great company and I am grateful for all support given by family, friends and co-workers. Finally a special thanks to my girlfriend Zhu Biyun for having faith in me during the whole project.

Tampere, 3.8.2016

Ilkka Veima

TABLE OF CONTENTS

LIST OF ABBREVIATIONS AND SYMBOLS

API	Application programming interface
AT	Assistive Technolgy
AWS	Amazon Web Services
BaaS	Backend as a Service
CaaS	Container as a Service
CMS	Content Management System
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DB	Database
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IaaS	Infrastructure as a Service
IoT	Internet of Things
JK	Jalkapalloklubi (football club)
MBaaS	Mobile Backend as a Service
PaaS	Platform as a Service
SaaS	Software as a Service
RAM	Random Access Memory
SDK	Sofware Developemnt Kit
WCM	Web Content Management system
wysiwyg	What You See Is What You Get

1. INTRODUCTION

The purpose of this thesis is to improve the functional and visual aspects as well as the maintainability of the Roxbury JK website. The website is made responsive for mobile users. Integration with social media services is improved. New data structures are added to enable collection of statistics and tracking of player/member information. Support for multiple teams is added. The solution should enable all content editing to be done without the help of the administrator. Making user interfaces improvements should be easier.

1.1 Background

Roxbury JK is football club formed by a group of friends who met during their time in Tampere Arts-Oriented Senior Secondary School [141]. After the matricular examination spring 2006, the common interest to football kept the group together. On July 2009 the Roxbury JK football club was founded. Since then the club has served as a way of bringing interesting people together in all sorts of events. From the small founding group the club has grown into an organisation of two football teams with over fifty members. There are two main reasons for creating and maintaining the Roxbury JK website: The first is to display the members and activities of the club, the second is the interest to web technologies by one of the founding members and the writer of this thesis.

1.2 Content management system

A content management system (CMS) is used to manage content. Managing content includes creating, storing, indexing, archiving, publishing and distributing content [75]. A web content management system is a CMS that is used to manage content in the web [1, 76]. The most popular content management system is currently WordPress [15, 142]. The WordPress platform was initially developed to be a blogging

platform, but has since evolved to a general website platform [151]. Other popular CMS platforms are Joomla and Drupal [142, 89]. WordPress, Joomla and Drupal are all web content management platforms. For simplicity web content management systems are sometimes referred as content management systems. In this thesis the word web is also omitted and CMS refers to a web content management system. There are four main features in a CMS: view templates, tags, administrator panel and data storage [90].

In a traditional website each URL points to a static HTML-file. The URL is used to access that specific file. In a CMS website a HTML-file is dynamically created and returned to the user. Each URL points to specific content. That content is stored in the database without any presentation style. The content is combined with a page template to create the HTML-file. [90].

The basic entity of a CMS is a web page where content is displayed. In a CMS a page template defines which content elements are displayed on the page i.e. the structure and presentation style of the page. The page template does not define the content on the page. There are two basic content element types: Text and multimedia file. [91]

1.3 Current solution

The purpose of the website is to record the history of the football club. History in this context means descriptions of the events and happenings of the club and its members. The website will over time accumulate content that will tell the story of the club. A CMS enables the creation of required content: stories, match reports, images and video. Basically the current solution is a blog running on the WordPress CMS. The WordPress installation has been customised with plugins, a little database customisation and theming. The plugins are used to link content from social media services. The basic WordPress blog database has been extended to store the player profiles of the members of the club.

The club memebers take turns in writing the match reports after each match. The match report is created by writing and publishing an article using the WordPress CMS. The player profiles are written in pairs. The members in a pair write a description of each other and the administrator will upload them to the website. The administrator is also responsible for keeping the WordPress installation and

the plugins updated.

Currently the database extension for player profiles does not have a user interface where an editor could modify the player information. The player database can only be modified by editing the database directly using the DirectAdmin web control panel provided by the hosting service. Using the DirectAdmin control panel requires significant knowledge of the system and is currently only used by the administrator. Player statistics are recorded manually in the match reports, but the data is not linked to the player information in the database. In the current solution it is not possible to differentiate players or events between the two or more teams.

Custom theme was created to promote the colors of the club and to display the data from the extended database storage. The custom theme is design for desktop users only and lacks any support for devices with smaller displays. All the customisation on top of the WordPress platform were done quick-and-dirty due to lack of time from the administrator. It is difficult to make any changes to theme and generally developing the solution difficult. The current solution therefore is devious to use and does not encourage the users to create new content. Without continuous generation of new content the solution will not serve as the history for the club and will become obsolete.

2. EXTERNAL SYSTEMS

There two notable system that are used in the club administration: osallistujat.com attendance service [107] and pelipaikka league management service [109]. The league management service is provided by the league organiser Palloliitto. The attendance service is used inform the club members of events and to ensure that there are enough players for matches and training sessions. Addition to training and matches, attendance for recreational events are also tracked using the service. Each memeber has an account to the attendance service and will use it to sign up for the club events.

League management system is used to sign the club for the league and ensure that all the players have the required licence. Start of 2015 Palloliitto launched a new league management system, TASO [108]. The new sevice is used for league sign up and player licences. As a new feature, TASO provides an application programming interface (API). Using the API it is possible to fetch the league table, match and player data for the club. The API, however, only provides preformatted HTML, which can be customised using cascading style sheets (CSS). That means the layout of the available data can not be changed.

3. REQUIREMENTS

The solution will act as content management system for the football club. It can be described as a football club blog with tracking of player statistics. The solution consists of three parts: users, content management and hosting. The users act on the website by creating, modifying and consuming the content. The website is accessed using a web browser. The host stores the content of the website and makes the website accessible through an internet connection.

3.1 User requirements

There are three different user types: Administrators, editors and readers. The administrators maintain and develop the solution, the editors generate and modify the content and the readers consume the content. The user roles are hierarchical so any action available to a reader is also available to content creator and administrator. Also any action available to content creator is also available for the administrator. The user roles are displayed in the table 3.1. The readers are mainly friends, family and opponents. The players will act as both readers and editors. Members with required technical skills act as administrators.

Table 3.1 *User roles*

Hierarchy	**User role**	**Description**
1	Administrator	Maintains and develops the website.
2	Editor	Creates and modifies the content of the website.
3	Reader	Consumes the content of the website.

3.2 Functional requirements

User actions as presented as functional requirements. Readers can view the content: teams, seasons, match reports and player profiles. Readers can sort and filter the

content they are viewing. For example a reader can select to view only players of one team instead of all players in all the teams. Readers can view player profiles that show statistics of the player. Readers can choose to view statistics of a player for all time, current season or a specific season. Editors should be able to create content without significant or any programing skills, but knowledge of a markup language, HTML or other, may be required. Editors can create teams, seasons, players and match reports. When an editor creates a match report it should be trivial for them to select players that scored, assisted a goal or were booked. It should be easy for the editors to add images and other media from social media services into the content they are creating. The functional requirements are listed in table 3.2

Table 3.2 *Functional requirements*

User role	**Action**
Reader	View the website with a browser. Register to the system to become an editor. View all players of the club. View all players of a team. View all match reports in a season. View statistics of a player (all time). View statistics of a player per season.
Editor	Login/logout. Create/upload new content (text/media). Link media from social media services. Create a new team. Create a new season. Create a new player. Create a new match report. Edit teams, seasons, players and match reports. Link match report to a season. Link match report to a team. Link team to a season. Link player to a team. Can add sponsor advertisements on the website.
Administrator	Update the system. Add new user to the system. Change user roles. Install plugins. Create a system backup. Can build a theme for the website. Can select a theme for the website.

3.2.1 Content types

Five content types are used to store the club information: team, player, season, match and player statistics. Each content type consists of data elements. A data element contains data. Data element details are given in table 3.3. Player email and phone number are used by the team managers to contact the players of team. Team managers are the members running the club and a team manager can be a reader, an editor and/or an administrator of the website.

3.3 System requirements

System requirements describe the general requirements for the solution. These are metrics that apply to the solution as a whole.

3.3.1 Accessibility

Majority of digital media is consumed using a mobile device [18] also other mobile use is overtaking desktop use. For example half of eCommerce transactions are made using a mobile device [93]. Following this trend the users should be able to use the solution completely with a mobile device. The users should be able to complete any reader and editor action using a mobile device The website should also be accessible through our domain roxburyjk.fi. Support for assistive technologies (AT) [8] is not required. Support for AT may be required in the future.

3.3.2 Storage

Current web hosting only offers 100MB of storage space. It has been enough for about five years (2009-2014), but has been limiting the content generation for over a year now. A few gigabytes of storage is more appropriate to keep the solution running for another five to ten years. There is growing need to upload media content, but most of it can be stored in various social media services, such as instagram, facebook and youtube. The more media is uploaded to social media services, the less storage space is required.

***Table** 3.3 Content types*

Content type	Element	Element description
Team	Name	Name of the team.
	Description	Description of the team.
	Players	The players in this team. References the player content type.
Player	Name	Name of the player.
	Number	Number of the player.
	Email	Contact email address. Used by team managers. Not displayed on the public website.
	Phone number	Contact number. Used by team managers. Not displayed on the public website.
	Image	Player profile picture.
	Description	Description of the player.
	Position	Players favourite pitch position.
Season	Name	Name of the season.
	Start date	Start date of the season.
	End date	End date of the season.
	Description	Description of the season.
	Teams	Teams related to the season. Refrences the team content type.
Match	Title	Title of the match.
	Date	Date of the match.
	Season	References season content type.
	Description	Match description.
	Players	Players participating in the match. References the player content type.
Player statistics	Type	Statistics type: Goal, assist, booking.
	Player	Related player. References the player content type.
	Match	Related match. References the match content type.
	Time	Time of the statistic event.

3.3.3 Performance and user experience

Performance is not a main concern as the website expects little traffic. Figure 3.1 shows page loads per month of each year and the total page loads of each year. Figure 3.2 shows page loads per month as a histogram. Additionally figure 3.2 shows the total of all page loads, total of comments and the number of all time high pageloads, which tells maximum of pageloads per one day.

Months and years

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Total
2012								806	403	336	302	84	6 540
2013	25	904	962	741	915	670	956	621	573	49	42	82	6 422
2014	76	110	1 010	745	813	804	588	618	1 006	183	133	336	3 016
2015	200	185	310	322	319	557	372	243	242	122	49	95	1 823
2016	122	76	110	351	655	385	124						1 931

Figure 3.1 *Current solution traffic overview - pageloads per month and year*

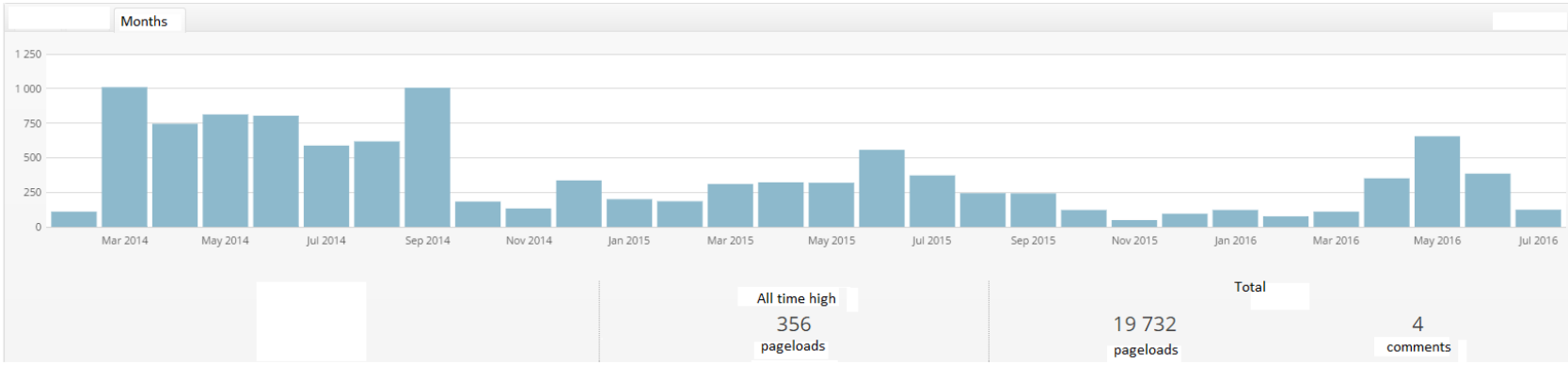

Figure 3.2 *Current solution recent traffic - pageloads per month*

Based on the traffic shown in the figures 3.1 and 3.2 it is clear that the performance requirements are very low. Performance testing should still be applied to check that the solution can handle expected peak loads. The expected peak load derived from the statistics is less than 500 page loads per day. The solution is considered powerful enough if the end users find it reachable, responsive and fast enough. The website should load fast enough, it should react to user inputs fast enough and should work on both desktop and mobile browsers so that the end user is satisfied.

3.3.4 Maintenance

Limited administrator time is what has kept the website from developing faster. Maintenance work should be as automated as possible. Automation of repetitive tasks allow the administrator to spend more time in developing the website. All content generation work should be possible for editors to complete without the help of the administrator. Administrators work should only consists of keeping the solution up to date and development of new features. From time to time there will also be requests by the content creators to change the look of the website. Modifying the theme should be made as easy as possible.

3.3.5 Budget

The website is expected to have low traffic (see 3.3.3) and therefore can not justify high costs. Current WordPress based solution is hosted at Zoner web hosting for 66,96€/year, 5.58€/month. Chosen solution should be possible to be hosted for similar costs. The costs should be minimised, but 10€/month (120€/year) is still reasonable. Split among roughly forty (40) members that would be 3€/year/member and easily justified for the members of the club. More content will be created over time and the demand for storage capacity will grow. The costs will therefore grow over time and the costs should be as low as possible in the beginning. The budget is an important requirement, but low maintenance is still prioritised as long as the costs stay reasonable. It would be difficult to justify the costs of over 5€/year/member (200€/year).

3.4 Other requirements

The current hosting provider provides mailing lists and dropbox is used to share files. There is no need to change these, but these should be considered if the hosting provider changes.

4. IMPLEMENTATION ALTERNATIVES

The solution is divided into two parts: the system and the host. The system is the software that enables the users to view, modify and the store the content. The host makes the system available for the users. The host includes the hardware and the networking that makes the system accessible for the users.

4.1 System implementation alternatives

Three different methods to implement the system are considered. An available system or a content management system can be used or a custom solution built. The available services and content management systems are evaluated against the requirements to see if they can provide what is required. Different implementation approaches are considered to find out the best way to build the custom solution. The best alternatives of each method are compared to decide the best way of implementing the system.

4.1.1 Available solutions

In this section we will take a look at available services for football or sport teams website hosting. Only services in either finnish or english are considered. The service must also operate using the SaaS -business model for easy deployment, use and maintenance.

Club Website

Club Website (`http://www.clubwebsite.co.uk/`) is a UK based service to create and host football club websites. The service has been in operation since July 2005 [145]. A demo site is availabe at `http://www.clubwebsite.co.uk/parksidefc`.

The front page shows a logo, navigation bar, image carousel and a collection of widgets. Different wigets range from social media and league table to polls and weather. The demo site theme is not responsive, but the mobile friendly version is available at `http://m.clubwebsite.co.uk/parksidefc/Home/login_home`.

The Club Website service is free to use, but they reserve the right to change the charges at any time. If the charges change they promise a notice at least 30 days in advance [146]. The Club Website makes money with optional services that cost money and by putting advertisements on the clients websites.

The Club Website terms of service section 2.3 states that: "Club Website has full permission and rights to advertise and generate revenue from advertising on your website. If necessary, you will allow Club Website to reposition the content on your website in order to accommodate any 3rd party advertising. Club Website reserves the right to append a hyperlink to their own website at the very end of each page designed for you." In section 2.4 it also allows the governing football association to display messages and other communications on a designated area on the website. [146]

The Club Website service has the required features and it is free to use. There were no information on how much storage space is offered. Although the terms of service (5.1) state that "There are no catches or hidden costs associated with this service.", more storage could be offered as an optional service after a predefined limit. The service is made easy to use for non-technical users and enables only limited theming. You can change the layout, choose which widgets to show and select your club colours to be used, but there is no option to provide a complete custom theme. It is not possible to fully control how the club website will look.

The account creation procedure was simple. The customer is prompted for the name of the club, club colours, website URL (clubwebsite.com/yourclubname) and contact details. After creating the account the website is generated and can be accessed by the chosen URL. A verification of the account is required via email. After verification it should be possible to log into the administration panel of the website, but it was not possible at this time. It could be that the service is going through changes and is not fully operational since the company was recently acquired by teamer (another service provider 4.1.1) [100].

teamsnap

Teamsnap (`https://www.teamsnap.com/`) is a team management service. The service provides contact information management, a private message board, email, event schduling and sms services [130]. The contact information can be extended with custom fields [133]. The service also includes an instant messaging system called TeamSnap Live [135]. A mobile app is provided to use TeamSnap Live. Scheduling is also available on both the Web and the mobile app. The manager can create events for the team. A location and a Google maps link can be added to the event, which should help everyone to find the right place at the right time. It is also possible to synchronize the events into your own calendar [134]. Teamsnap can be used as an attendance service. Members of the team can set their availability for the events: Attend, will not attend, not sure [129]. Teamsnap will also track payments [131]. There is a limited free membership. Other memberships start from $7.99/month [132].

The target audience of teamsnap seems to be coaches and managers of teams. It is a central place to manage a team, but most of the features can be replaced with other services. You can set up instant messaging groups, shared calendar and there are attendance services like `osallistujat.com` (2). Teamsnap will make it much easier as all functions are controlled from a single administrator interface, but the service does not offer content management.

website4sport

Website4sport (`http://www.website4sport.com/`) is website and hosting service for any sports club. Pricing plans display annual costs and starts from $54.99/year for a single team. The most expensive plan costs $154.99/year and includes all features. Even the most expensive plan only offers 360MB of storage. [148]

Website4sport offers customizable website with event scheduling and support for statistics. The service includes player biographies and photo gallery with unlimited photo uploads. Website4sport also offers additional customization service to make the website just as you want it. [147]

Website4sport seems to have basic features for club management and while custom

themes can not be created, they offer to customize your website beyond the controls available in the administrator interface. The service does not offer enough features for the price and the limited storage is not sufficient.

sportsclubservices

Sports Club Management Services website (`http://www.sportsclubservices.com/`) is simple. The front page only loads the HTML-document and the header image. There are no javascript or CSS -files. The name implies that it is a system directed at any sports club, but in the description it becomes volleyball club management system, dedicated to volleyball clubs. The system is designed to be a management service, not a content management service. The service enables the volleyball club manager to handle player registration, player billing, group emails, event scheduling and player attendance. [120]

The website is pure HTML without any CSS or javascript, which makes it look simple and outdated. All page changes create a HTTP request, but as the returned documents are small, the system is fast. The service also includes player information pages, but it can not be considered a content management system. To use the service the club needs a website: "All you need to use our service is your club's website. We will work with your webmaster to establish the necessary links to get the system operational." [120]. It is not documented what the necessary links are.

The pricing is $25/team/month for clubs with fewer than ten (10) teams and $150/month + $10/team/month for clubs with ten (10) or more teams. The fees are not updated since 2010. [121]

Pitchero

Pitchero (`http://www.pitchero.com/`) is a sports club website creation and hosting service. No programming or HTML knowledge is required, which means that the customization of the website is limited to the controls provided in the administrator interface. The service can be used with any device and provides unlimited storage. Player information and payments are managed from the administrator interface. [114]

After setting up an account a website is created for the customer. In the administrator interface the user can manage content, such as posts, news, photos, polls, videos, documents and sponsors. The administrator can create multiple teams and add players, events and fixtures for that team. For each fixture the user can assign a squad, write a report and set match statistics. The player information include: name, pitch position, previous clubs, contact details and description. The administrator can send group messages and the audience is easy to filter by team, role or individually with the given tools. Website performance monitoring, eCommerce shop and fundraising tools are also included. The layout and colors can be changed from the administrator interface. The layout is chosen from predefined templates and primary and secondary colours can be changed.

The free Basic Plan is limited to three teams. Starter plan (10€/month) [114] adds more features and removes the full page advertisements. Standard plan (30€/month) [114] adds more templates to choose from and includes a request for the pitchero design team to build a custom theme. The custom theme is based on four predefined styles: Super Sharp, Pride & Glory, The Classic and The Old Boy [113]. The standard plan also adds google analytics, removes sidebar advertising and gives option to use a custom domain name. With pro plan (60€/month) [114] you can opt-out from partner promotions and remove video advertising. The pro plan adds PayPal shop which can be used to sell club merchandise. The package information is only available during and after creating an account.

LeagueRepublic

LeagueRepublic `www.leaguerepublic.com/` provides league and club management services. LeagueRepublic can be used to host leagues and clubs. LeagueRepublic clubs can be signed up to the leagues. Only the club management is considered as there is no need for league management. The website uses site builder, which makes it easy to build a website, but limits the customisation options. Customisation includes changing content, images and colours. There are no limits for teams and players that can be included in the club. Hosting and storage are provided for free and there are no storage limits. The services responsive design enables access with any device. Support is offered free of charge. [86] After logging in the administrator can create teams, players, matches and match reports. For each match the score can be set, but the statistics can not be assigned to individual players.

LeagueRepublic offers website integration through json API, which can be used to build a custom website that takes advantage of the LeagueRepublic management systems. [85] The JSON Web Services (json API) costs 250€/month [87]. The club management service is free to use, but using a custom domain name costs 100€/year. Removing advertisements from the website costs 250€/year [87].

teamer

Teamer `teamer.net` is a free to use sports team management service. Teamer is a service to manage player information of the team. The service enables easy and efficient communication to all members. The service is also used as an availability or attendance service to see who is available for the scheduled events. Relevant documents can be uploaded and shared using teamer. [127]

Teamer allows teams to collect payments from the members. The only cost is 2.4% + 20p per transaction. [128] After creating an account the administrator can create multiple teams, but there are no links between the teams. The team page is not visible to public i.e. only members can view the team page after login.

Availabe solutions summary

There were multiple services available and most of them are free to use or operate using the freemium model.

> "The word **freemium** is a combination of the words **free** and **premium**.
>
> It describes a business model in which you give a core product away for free to a large group of users and sell premium products to a smaller fraction of this user base." [57]

The services profit comes from the advertisements included in the customer websites. Most of the available services, however, were focused on club or team management features and not in content management. Table 4.1 presents the available solutions and the required features. The data for the table is based on what can be found on the websites of the services. The best available solution are the Club Website (4.1.1) and Pitchero (4.1.1).

Table 4.1 *Availble services and required features*

Service	**Price /month**	**Storage**	**Multiple teams**	**Players**	**Seasons**	**Match reports**	**Statistics**	**Custom theme**	**Responsive design**
Club Website	free	?	✔	✔	✔	✔	✔	✖	✖
teamsnap	$7.99	0	✖	✖	✖	✖	✖	✖	✔
website4sport	$4.58	max 360MB	✔	✔	✖	✖	✔	✔	✖
sportsclubservices	$25	?	✔	✔	✖	✖	✖	✖	✖
Pitchero	free	?	✔	✔	✔	✔	✔	✖	✔
LeagueRepublic	free	unlimited	✔	✔	✖	✔	✖	✖	✔
teamer	transaction based	?	✔	✖	✖	✖	✖	✖	✖

The Club Website is free and it has all the content features required from the website. The only downside of the Club Website service is that theming is not supported. The Club Website also forces advertisements on the site, which could be why theming is not supported. It would be difficult to force advertisements on a custom theme. There is not information if custom domain can be used with the service, but that could be ignored as all other requirements are fulfilled. The service will not be used however, because at the moment is does not work. The account could not be verified and the website would not go live. It could because of the recent acquisition by teamer and because of that, it is not clear if the service will continue as described or when the service will be operational again.

Pitchero also has all the features needed for the solution. The downside, as with the Club Website, is the limited theming. To use a custom domain however the free plan has to be updated to the standard plan (30€/month). The annual cost would be 360€, which is five times more than the current solution and with the standard plan the website still includes advertisements.

4.1.2 General configurable solutions

A general configurable system is a system that can be used to build systems with varying requirements. A general configurable system has predefined set of configurable parameters. Different requirements can be fullfilled by changing the values of those parameters. As an example the system could have an option to turn communication encryption on or off. More advanced configuration can be achieved by a plugin system. The plugins can implement complex features that are enabled by installing the plugin.

Content management system (CMS) is a type of configurable system and is one option to implement the system. Content management systems often have plugin system with multiple plugins to choose from. Basic features of a CMS enable the users to create and manage the content from the administrator panel [92]. Beyond the basic features, the CMS should be configurable or otherwise extendable (without programming knowledge) to address the football club specific requirements. The requirements include the player database, adding statistics to the match reports and maintenance of multiple teams. The requirements are discussed in chapter 3.

Three most popular CMS platforms are considered as they are most likely to have

enough features to satisfy all the requirements and because it would take too much effort to go through all available content management systems. Wikipedia lists over a hundred content management systems [149] and CMS Critic search finds 201 content management systems [42]. The three most popular content management systems are WordPress, Drupal and Joomla [142, 21].

Wordpress

WordPress is an open source content management system. It is the market leader and powers roughly 25% of all websites and over 50% of websites using a CMS [143, 116]. Additonally wordpress.com (commercial service by automattic) offers free WordPress hosting [152]. There are two plugins that might satisfy the requirements: WP Club Manager and SportsPress. Both plugins have good reviews and multiple installations.

WP Club Manager is free WordPress plugin designed to manage a sports club website. The plugin supports multiple sports, including football. There are over 1000 active installs of the plugin [39].

The plugin author offers a free theme and additional features are available for purchase. The additional features are: Sponsors pro, players gallery, player appearances and score summary [33]. Sponsors pro adds functions to help attract more sponsors. These functions include targeted advertisements and reports, which can be used to justify investment of the sponsors. Sponsors pro add-on has annual cost of £10 [37]. The players gallery plugin adds a player gallery and has an annual fee of £6 [34]. The player appearances plugin adds statistics of match appearances to the player profile page and has annual fee of £6 [32]. Score summary plugin adds more result details in the match page, mainly the half-time score. The score summary plugin is free to use [35]. The additional features are licensed annually. Premium theme that is build to be used with the free plugin has an annual license fee of £29 [36]. Most support requests for the plugin have been resolved [38] and most of the unresolved ones are requests for making the plugin support a new sport.

To satisfy the requirements with WP Club Manager the theme, player gallery plugin and player appearances plugin are needed. The pro version of sponsors is not required. This would make the total annual cost £41 (roughly 55.4€ at current rates). The author of these plugins offers a 50% discount for renewing a licence af-

ter the first year [40]. After the first year the expense would be £20.5/year, around £1.7/month (roughly 2.3€/month at current rates). These expenses would be on top of the hosting service fee. The expenses are reasonable, but the outlook of the site would be limited as there is only one custom theme available at this time. As the core plugin is free to use a theme could be build to use with WP Club Manager.

SportsPress is another plugin for the wordpress platform. SportsPress plugin extends the WordPress platform so that sports clubs can manage their sports specific content. "Features a suite of sports tools including fixtures, results, automated standings, players rankings, and individual profiles for clubs, players, and staff." [136]. SportsPress plugin is free to use, but there is also a pro-version. The pro-version adds more customisation options, sponsor space, sponsor management and a league menu. The minimum fee for the pro-version is $99/year. That is $8.25/month or 7.6€/month (with current rates). A finnish translation has been made for the SportsPress plugin, but that would affect only the administrator panel. All our players are fluent in english, so the translation, while nice to have, is not required. Plugin author provides a simple theme, called rookie, for free and it is designed to work with the plugin [139]. There are also premium themes starting from $79/year [137]. The pro-plugin and the theme licence has to be renewed annually, but after the first year a 50% discount is given [138, 137]. There are over 10 000 active installs of the SportsPress -plugin [136].

Using the free version for the plugin and the free rookie -theme it would be possible to satisfy the functional requirements. A custom theme could be built and used with the SportsPress plugin. The pro-version of the plugin and the premium themes are over the budget.

Drupal

Drupal is an open source content management software. When compared to WordPress and Joomla, the key differentiator for Drupal is flexibility. While Drupal has many core features, the flexibility comes from the modularity. The Drupal core can be extended with modules [48]. Many CMS comparisons consider Drupal the most extensive and capable of building the most advanced sites. Compared to WordPress and Joomla it is also considered to require the most technical expertise. [126, 125]

Only a few modules related to sports club management are available [49]. The most

promising one, titled Sports Club Management, is not in development anymore and is minimally supported [140]. The usage statistics page for the module shows a steady decline in usage [51]. While the usage statistics are not comprehensive, the usage trend and activity on the web page imply that new users are not adopting this module. There is also a key feature, theme customisation, missing.

While the modular nature of Drupal makes it easy to reuse features, it also requires more technical knowledge, as mentioned in many CMS comparisons. The complexity would probably result in more training and support required by the editors of the Roxbury JK website.

There is a "One League API" initiative for Drupal 7. The goal of the initiative is to develop a common data model and related APIs for league-related entities and operations; in order to help reduce module duplication and facilitate interoperation between various league related modules [49]. There is a module based on the initiative called League.

The League module has the required functionality from creating teams and players to adding match statistics [122]. The League module is free to use. There is a theme built for the League module [52]. From the source it is not clear if this theme is free to use or not. From the tutorials [66] the administrator interface does look outdated. It is would be too much work to build a custom theme and customise the admin interface. The League modules usage statistics show a steady rise for the past 18 months and there are over 100 module installations [50].

While rich in features, Drupal with the League module seems difficult to use. The user interfaces do not look polished, which will make it off putting for regular users. There are not many users (compared to the WordPress plugins) for Drupal and the League module and so there will likely be less support. Considering the budget, Drupal is a valid option as there are no extra fees after the hosting service.

Joomla

Joomla is an open source content management system [78]. Joomla is considered a compromise between WordPress and Drupal as it is more powerful than WordPress, yet easier to use than Drupal [126, 125]. There are a lot of features included in the core of Joomla. For example being multilingual by default, frontend edit-

ing (for small changes no need to login to the admin panel) and content versioning [77]. There are two extensions for football club management: JoomSport and JoomShapers Soccer template.

JoomSport is a free extension for Joomla CMS. Reviews of the extension give an overall score of 97/100. It is possible to create teams and players and add player statistics. [11] There is a demo site available at `http://demo2.joomsport.com/` (2016-01-21). The JoomSport extension has most of the required features, but it is not mobile friendly or responsive. This feature is saved for the paid pro-version of the extension [14]. The pro-version costs $158 for one time purchase that includes all updates for the future or an annual licence of $99, which includes updates for a year [14]. To have a responsive site an add-on is needed and it costs additional $49. There is no mention if the fee is one time only or an annual licence. [13] Themes built for the extension start from $38 [12].

The standard version of the JoomSport extension does not satisfy all the requirements as it can not be used with a mobile device. The cost of the pro-version with the responsiveness add-on and a theme totals $245 (around 227€ using current rates) assuming that all prices are one time only and the cheapest theme is chosen. If the solution would be used for the next five years without extra fees, the annual cost would be roughly 45€/year. Together with the hosting service fee this is over the budget.

Soccer template is a paid service from JoomSharper [80]. There is a demo site at `http://demo.joomshaper.com/2015/soccer/` (2016-01-21). The Soccer template includes a Joomla theme and a SP Soccer component. The SP soccer component brings the club management features [79]. To get the theme the customer has to join the Joomla templates club. The price for three month membership is $59 (53.74€at current rates). This membership includes one licence, which means that the use of the template is limited to one website. A member of the templates club can download and use all available templates and extensions. The templates installed can be used after the membership ends, but no further updates are available unless the membership is renewed [81]. The use of extensions that come with the installed template should be allowed after the membership expires, but it is not clear from the information on the website. Since the terms of use also allow the modification of the theme, it could be improved to fit the needs [81].

General configurable solutions summary

The most suitable alternatives looked at in this chapter are SportsPress (default version) for WordPress and Soccer template for Joomla. JoomSport looks promising, but is over the budget. League with Drupal has the features and does not come with extra fees, but requires more maintenance and the interfaces are not as polished as with the other alternatives. The WP Club Manager is a good option, but the licence terms are complicated. From the WordPress plugins SportsPress has ten times more users than WP Club Manager and it is backed up by more than one developer. The SportsPress pro-version is over the budget. The Soccer template is paid service, but much cheaper than the pro-version of SportsPress. The fee for Soccer template is closer to one time fee as the theme can be used after the membership expires. The membership could be renewed for every major Joomla update to get the latest features. Soccer template for Joomla is the most suitable alternative within budget. SportsPress has better adoption than the Soccer template and the default version is free to use, which makes it a valid option as well.

The content management systems and the required features are listed in table 4.2. Data for the table is based on the data available on the websites of the plugins. Prices are calculated from what would be required to meet the requirements. The table shows that almost all alternatives implement most of the features. The difference is in other features like price and customisation. Compared to available services, shown in table 4.1, general configurable CMS solutions have better support for the required features. The general configurable CMS solutions would require more maintenance work compared to the commercial services.

Table 4.2 *Availble CMS and required features*

CMS	**Plugin /extension**	**Price /month**	**Multiple teams**	**Players**	**Seasons**	**Match reports**	**Statistics**	**Custom theme**	**Responsive design**
WordPress	WP Club Manager	2.3€	✔	✔	✔	✔	✔	✔	✔
WordPress	Sports Press	7.6€	✔	✔	✔	✔	✔	✔	✔
Drupal	Sports Club Management	free	✔	✔	✔	✔	✔	✖	✖
Drupal	League	free	✔	✔	✔	✔	✔	✖	✖
Joomla	Joom Sport	3.75€	✔	✔	✔	✔	✔	✖	✔
Joomla	Soccer Template	17.92€	✔	✔	✔	✔	✔	✔	✔

4.1.3 Custom solution

One option is to build a new system for the football club content management. A backend as a service could be used to abstract the database and related operations, which would make the development faster.

Backend as a service

Backend as as service (BaaS) is cloud service model that aims to simplify the data access for an application by abstracting the whole server stack into an API [84]. It can also be described as a service to replace backend needs, such as data storage, data queries, data validation, data security, file storage, push notifications for mobile devices and analytics [23]. In some resources it is also labeled as MBaaS - mobile backend as a service [84]. Most of BaaS services are build on top of a cloud server farm and provide a web API.

In a normal use case a mobile application developer would use the BaaS to store the mobile application data. The mobile application would communicate with the backend service through the internet. This solution abstracts the server, server software and database maintenance from the mobile application developer and solves other problems such as load balancing. Most of the BaaS -services also offer features to speed up the application development. These features include user management, including OAuth, push-notifications to the mobile application and usage analytics. All of which take significant development effort. The purpose of a BaaS service is to speed up the development of the application as data storage and related tasks are provided by the BaaS provider. [65]

Cloud service models

Backend as a service is continuation for other cloud service models [124]. Infrastructure as a service (IaaS) is a service that provides a server or a whole data center. All hardware is owned by the service provider and used by the customer. This service model abstracts the infrastructure (data center) management. The customer gets all the infrastructure components (computing, networking, storage, security) in a managed facility. [53] Examples of IaaS providers include amazon web services, rackspace and bluelock [29].

Platform as a service (PaaS) is service that provides the operating system on top of the infrastructure [53]. This service model abstracts the infrastructure and the server management and the customer is only responsible for the server software. Examples of PaaS providers include amazon web services, google apps and salesforce.com [30].

Software as a service (SaaS) is a service that offers everything that is needed to run software [53]. This service model abstracts the infrastructure, the server management and the software management (installation, updates, development) and the user can just use the software. Examples of SaaS providers include Cloud9 analytics, Oracle on demand CRM [31] and wordpress.com [152].

BaaS is a service that abstracts the whole server stack [84]. BaaS can be considered a specific type of SaaS that is targeted for the application developers. Instead of an end user using the software, the application developer uses the backend service to create an application. It is a ready solution that is used as is and the application developers only concern is the application they are developing [124].

Backend as a service providers

Three open sources BaaS alternatives and eight commercial BaaS providers are available. All three open source solutions were licenced with Apache 2.0 and can be deployed freely. Most of the eight commercial solutions provided a free starter or a limited account or were free to use in an open source project. There were few solutions for which pricing details were not easily accessible. Commercial BaaS provider details are listed in table 4.3 and open source BaaS provider details in table 4.4. Along with pricing information the tables show if the solution includes private deployment, push notifications and social login. There are also notes on the solution and the key differentiator. Private depoloyment means that the solution can be hosted in private servers. Private deployment offer some independence from the service provider. Push notifications enable notifications to be sent to mobile devices using the solution. Push notifications are not required for this system, but it is an important feature of a MBaaS. Social login enables user to register and login using a social platform like facebook or twitter. Social login feature should have significant impact on development time of the system.

Table 4.3 *Commercial backend as a service providers*

Table 4.3 *Commercial backend as a service providers*

Name	**kinvey**
Pricing	Free starter plan, full service $24K/year
Website	`http://www.kinvey.com/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	Premium service with most features
Notes	The service seems comprehensive. It is aimed at enterprises and has reduced service for startups and individual developers.
Name	**kii**
Pricing	Not available
Website	`https://en.kii.com/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	China services. Focus on IoT (Internet of Things)
Notes	-
Name	**built.io**
Pricing	Free limited plan. $99/month, $399/month (2015-08-07, prices no longer available)
Website	`https://www.built.io/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	Premium service. Bunch of features.
Notes	Free webapp hosting included. Comprehensive documentation. Student account.

Table 4.3 *Commercial backend as a service providers*

Name	**anypresence**
Pricing	Not available
Website	`http://www.anypresence.com/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	No vendor lock-in due as source code released if necessary. Premium service.
Notes	"Enterprise backend server", which could do as the web app, but not sure if included in free plan.
Name	**Sencha Touch**
Pricing	Free licence for open source projects. Commercial licence.
Website	`http://www.sencha.com/products/touch/`
Private deployment	✖
Push notifications	✖
Social login	✖
Differentiator	Sencha Touch platform converts HTML5 applications into native mobile applications.
Notes	Not building mobile app, but a website. Not actual BaaS provider, although there is Sencha Space.
Name	**appcelerator**
Pricing	Not available
Website	`http://www.appcelerator.com/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	Converts Web apps into native mobile apps. Comprehensive analytics.
Notes	Enterprise oriented

Table 4.3 *Commercial backend as a service providers*

Name	**parse**
Pricing	Free plan, price based on requests/s.
Website	`https://parse.com/`
Private deployment	✖
Push notifications	✔
Social login	✔
Differentiator	Advanced analytics and targeting.
Notes	Pricing is too unpredictable.
Name	**apiomat**
Pricing	Includes free account. Monthly subscription.
Website	`http://www.apiomat.com/`
Private deployment	✖
Push notifications	✔
Social login	✔
Differentiator	-
Notes	Not as finished as other products.
Name	**Openshift**
Pricing	Free plan available. Base price $20/month
Website	`https://www.openshift.com/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	Red Hat backed.
Notes	Free plan is available world wide, but small servers only in US. The free plan includes only 3 small servers.

Most commercial BaaS platforms in table 4.3 are full featured, but the price is too hight, the pricing is too unpredictable or the pricing details are not available. Most of them seem to target enterprise customers instead of small teams, startups or individual developers. There are few platforms with a different focus: Kii [83] is targeting the internet of things (IoT) domain and Sencha Touch [119] converts HTML5 applications into native mobile applications. Anypresence [3] is trying to stand out by promising no vendor lock-in.

> "Vendor lock-in is a situation in which a customer using a product or service cannot easily transition to a competitor's product or service. Vendor lock-in is usually the result of proprietary technologies that are incompatible with those of competitors. However, it can also be caused by inefficient processes or contract constraints, among other things." [118] Anypresence promises complete access and control over all run-time application source code [4].
>
> "*Vendor lock-in*, or just *lock-in*, is the situation in which customers are dependent on a single manufacturer or supplier for some product (i.e., a good or service), or products, and cannot move to another vendor without substantial costs and/or inconvenience." [115]

"The fear of vendor lock-in is often cited as a major impediment to cloud service adoption." [118]. The customer often becomes dependant on the service provider and any problems in the service have direct consecuences for the customer. Even if the service has significant problems it might practically impossible or at least very costly to change the service provider. More on this in section 4.1.3.

The most interesting commercial BaaS provider is Built.io as they have free student plan available [24] and a possibility to host web application [25]. The student account is free for one year, but further cost details are not available. Additionally the cost of the added web application hosting is not available.

Table 4.4 *Open source backend as a service solutions*

Name	**Usergrid**
Pricing	Free
Licence	Apache 2.0
Website	`http://usergrid.apache.org/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	Open source
Notes	Backed by Apache Software Foundation.

Table 4.4 *Open source backend as a service solutions*

Name	**Deployed**
Pricing	Free
Licence	Apache 2.0
Website	`http://deployd.com/`
Private deployment	✔
Push notifications	✖
Social login	✔
Differentiator	Open source, easily synchronize data across different users.
Notes	Good first impression from web application point of view.
Name	**Baasbox**
Pricing	Free
Licence	Apache 2.0
Website	`http://www.baasbox.com/`
Private deployment	✔
Push notifications	✔
Social login	✔
Differentiator	Open source, also offering a hosting service.
Notes	Looks to have most features of the open source alternatives.

Usergrid has been adopted by Apache software foundation 2015-08-19 [72]. Usergrid has been open source since 2011 and is used by a number of big companies. The platform is full featured and has software development kits (SDK) for several technologies. [6] Usergrid looks to be a mature platform and being open source it is possible to deploy anywhere. The platform is also well documented [5]. The github statistics show that is is still actively developed [64].

Deployd is a backend service for building an API [43]. From the BaaS -providers looked at, the ones which do not have push-notifications are Deployed and Sencha Touch. Instead of beign backend service for mobile apps, Deployed is more general purpose solution for accessing and managing data using a web API. The github statistics show that Deployed is still beign developed, but the main development effort has been in 2013 [61].

Baasbox seems to have most features of the three open source services. The features include push notifications, social login, content and file management and database maintenance interface [10]. Baasbox offers a paid hosting service, but the prices are based on requests/s [9]. The costs for the customer are difficult to predict when this pricing model is used. The github statistics show that it has been actively developed, but the past six months show only one commit [59].

From the open source BaaS providers in table 4.4, Deployed is the most interesting option for a web application backend as it does not target mobile applications directly. Usergrid looks the most reliable as it is backed by Apache and the development is still active. The advantage of Baasbox is the hosting service, but the complex pricing model makes it unattractive.

Using backend as a service

Using any BaaS would speed up the development of a custom solution [82]. There are open source services and commercial services using freemium (4.1.1) model, which helps staying in budget. Building a new custom solution would still require significant amount of work, even if, the backend is abstracted to require minimum development. Problem with the open source approach is the need to handle deployment and maintenance. From the open source solutions only BaasBox offers a hosting service, that is not for free of course. Commercial providers all provide a hosting service, but some have a difficult pricing model that is based on HTTP requests or API calls per second. It would be difficult to calculate or predict the monthly price using such pricing model. A monthly subscription model would offer more predictable costs. Although considering the low traffic expected, a pricing model based on HTTP or API requests could turn out cheap.

In this work the BaaS would be used with a web application instead of a mobile application. The BaaS -concept is targeting mainly mobile application developers. A mobile application is running on the users device, but the web application needs to be hosted on a server. There would be need to pay for two service providers (BaaS provider and hosting service provider) which would increase the cost and there would be maintenance work for both services.

Using a BaaS -provider will also introduce the problem of vendor locking (4.1.3). Once a service is selected it will difficult and/or expensive to change the service

[144]. All data will be stored in the service and extracting the data, in a format that can be reused, may be difficult. In this solution not just all the data will be in the service provider's servers, but as the solution would be developed using the service provider's BaaS-API it would require significant amount of work to switch the service provider (modify the solution to use some other service provider's API). So once a service provider is chosen it is not trivial to make a switch. Possible scenarios where the service provider change is needed include the change in the pricing of the service to something unacceptable, unexpected problems in the service or any other problem with the service provider that would make service inconvenient to use. [99, 117] As an example Parse (see table 4.3) was acuired by facebook and then shutdown [110].

Extending a content management system

A CMS provides most of the required functionality. A CMS can be extended to address all the requirements. Common ways of extending a CMS is by theming and plugins. Theming is used to customise the appearance of a blog or a website. The open source content management systems can also be modified directly. While theming is necessary to get the right look and feel it is not used to implement new content types or features. To add new type of content into a CMS three things are needed: Storage structures, control panel functions and display templates. New data storage structures are needed to store the new type of content. New control panel functions are needed to create and manage the new type of content. New templates are used to display the new type of content. This can be achieved by developing a plugin for the CMS. Using plugins to extend a system improves the design and makes the system more flexible and maintainable. Plugins are a good way to implement the strategy design pattern. Using plugins reduces dependencies and makes parallel development easier. [67]

Default content types in a content management system are the types needed for a blog: A page and a post. The developed plugin should provide the new content types required (3.2.1). A theme is needed to give the website the right look. There are hundreds of content management systems (4.1.2) and most of them can be extended with a plugin.

PHP is the dominant language for the web and ASP.NET, Java, Ruby and other technologies are powering more websites than node.js [22].

> "Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world." [103]
>
> "As an asynchronous event driven JavaScript runtime, Node is designed to build scalable network applications." [102]

Still only CMS's build with node.js are considered, because of the adavantage that only javascript knowledge is need for both front- and backend development. This way only one programming language has to be to learned. Learning only one programming language will save time and make the maintenance easier.

Table 4.5 lists content management systems build with Node.js. Not all solutions listed are mature and their development can be on hold. For example, Buckets statets that development is on hold [19], Calipso website is down and github statistics look like the project is not actively developed [60] and Camel and Hadron do not have a dedicated website, but only github and/or npm package pages. Apostrophe, Ghost, Keystone.js and Pencilblue are mostly described as more mature and having at least the basic CMS features (1.2). These are also listed in Node.js CMS comparison posts as the most promising solutions [20, 41, 153].

Table 4.5 *Node.js content management systems*

Name	Website	Description
Apostrophe	`http://apostrophenow.org/`	Apostrophe is a content management system that helps you stay on mission.
Buckets	`http://buckets.io/`	Open source CMS built on Node.js and MongoDB.
Calipso	`http://calip.so/`	Simple NodeJS content management system based on Express, Connect & Mongoose.
Camel	`https://github.com/cliss/camel`	A blogging platform written in Node.js. It is designed to be fast, simple, and lean.
Ghost	`https://ghost.org/`	A publishing platform for professional bloggers.
Hadron	`https://www.npmjs.com/package/hadron`	Minimalist blogging/publishing platform for Node.js.
Hatch.js	`http://hatchjs.com/`	Hatch.js - CMS without the 'meh' !
Keystone.js	`http://keystonejs.com/`	Node.js CMS & Web Application Platform.
Pencilblue	`https://pencilblue.org/`	Business Class Content Management For Node.js.
Taracot	`https://taracot.org/`	Pure JavaScript with flavour of MongoDB, Redis and Node.

Apostrophe is a design-driven, in-context CMS [7]. Compared to database-driven, where modifications are made to database usually in administrator interface and then displayed to the user, in design-driven the modifications are made on the same interface where to content is displayed. This approach is very user friendly and requires little technical knowledge, but it could make theme development more difficult. There are not many plugins available for Apostrophe.

Ghost is probaly the most mature Node.js CMS, but it is just a blogging platform [153]. There is a need to extend the CMS for more than just a blog and therefore Ghost is not a valid option.

Keystone.js and Pencilblue are considered more general purpose CMS [20] and github statistics show active development for both projects [62, 63]. The problem with Keystone.js is that there is no plugin system [153]. The plugin system for Pencilblue is at core of the system and all aspects of the system can be modified with a plugin [70]. Pencilblue comes out of the box with custom objects [26], that make creating new content types easy. Pencilblue is build to be responsive using Bootstrap, including the administration interface.

4.1.4 Selected implementation method

In this section the best alternatives of each implementation method are compared. A number of available solutions were found and the top two are Club Website and Pitchero. Club Website (4.1.1) is the best of the available services. The required content management is possible using Club Website and it is free to use. However theming is not possible with Club Website and the default theme is not mobile friendly or responsive. These could be overlooked as the theme may change to better in the future. Club Website is good, but not the perfect solution. The most important reason not to use Club Website as the implementation is that there is no information on the continuation of the service. Pitchero (4.1.1) was the second best of the available solutions. The required content management featurs are available and the theme is responsive, but it is not possible to use a custom theme. The most important reason to reject Pitchero is the budget. To use a custom domain, which is required, the standard plan has to be used instead of the free plan. The price of the standard plan is over the budget. The other available solutions did not satisfy the requirements (4.1.1).

Using a content management system did not have that many alternatives. SportsPress for WordPress and Soccer template for Joomla were the best ones (4.1.2). Both alternatives satisfy the functional requirements and are within budget. WordPress and Joomla are built with PHP. Hosting a PHP system should be cheap and easy as there is a lot of competition in PHP hosting. The downside of these implementation methods is that they are difficult to customise. The current solution is a custom system build on top of WordPress and the main reason to build the new solution is to make maintenance, usage and development easier. These systems (WordPress and Joomla) are also big, which makes them more difficult to understand when compared to smaller systems. Compared to a custom solution, it would be faster to implement the system by using the SportsPress or Soccer Template as they satisfy the requirements. The speed, however, comes at the cost of maintenance and further development. Looking forward a custom solution on top of a Node.js CMS is going to serve the developer and the users better. With Node.js CMS only javascript knowledge is needed, instead of javascript and PHP. In both cases knowledge of the CMS in question is required. The Node.js content management systems are easier to comprehend as their code base is smaller. To integrate with external systems (2) a through understanding of the system is required. When the solution is custom build the maintenance and development will be much easier later on.

Ten different Node.js content management systems (table 4.5) were considered as the basis of the custom impelementation. Only Keystone.js and Pencilblue were mature enough to satisfy the requirements. Pencilblue is preferred over Keystone.js because it fits the needs better. Keystone.js does not have a plugin system, but Pencilblue is designed to be extendable and the plugin system is at its core. Pencilblue also has the custom objects feature that will speed up the development. It is very easy to add custom content types using the Pencilblue custom objects. Pencilblue is built to be responsive, including the administration interface. As Pencilblue is open source it can be hosted anywhere. Finding a suitable hosting service should not be a problem even though there are more PHP than Node.js hosting services. A custom solution is selected as the implementation method and the solution is based on the Pencilblue CMS.

Backend as a service was considered as way of speeding up the development. Eleven BaaS providers were considered, but none had very useful features for web application development as the BaaS is mainly a concept for mobile application development. In the end a backend as a service was not needed as the Pencilblue CMS

already had most of the backend needs covered.

4.2 Hosting alternatives

Hosting alternatives can be divided into three groups: Private server, IaaS -providers (4.1.3) and dedicated web hosting. Private server is a computer in which all hardware and software are manually installed and a private internet connection is used. Private server requires too much maintenance and therefore is not an option. It would also require more money in the beginning and it could take years to pay back, when compared to hosting services [150].

IaaS would abstract the hardware maintenance, but not the operating system (OS) and database maintenance. IaaS providers include Amazon Web Service (AWS), Microsoft Azure, Rackspace, Digital Ocean and Linode. Of these Digital Ocean and Linode have to simplest pricing with a monthly subscription. Digital Ocean offers the cheapest plan $5/month: 512 MB RAM, 20 GB SSD storage and 1 TB transfer [46]. Dedicated web hosting provides all maintenance for the system and the customer is only responsible for the content. Dedicated web hosting would abstract all infrastructure maintenance, but would also be more expensive. There are also dedicated pencilblue hosting available. Pencilblue can be hosted with a service for Node.js application hosting. Two most important requirements for the hosting service are maintenance required and price. Node.js hosting services are listed in table 4.6.

Table 4.6 *Node.js application hosts*

Name	Storage	RAM	Transfer	Price	Notes	
even node	1 GB	512 MB	?	6€/month	Monthly subscription.	[54]
Nodegear	unlimited	512 MB	unlimited	$6/month	Price is $5/month for a production server +$1/month for database.	[101]
cloudnode	100 MB	?	1M/req	19€/month	Limited by requests/month.	[28]
OpenShift	1 GB	512 MB	?	0.02€/hour	3 small gears offered for free. Additional small gear 0.02€/hour.	[104]
Modulus	1 GB/files, 1 GB database	512 MB	?	$0.04/hour	1 GB database storage is $5/month. About $33.80/month in total.	[97]
Heroku	10K rows Postgres	512 MB	?	$7/month	Monthly subscription. MongoDB available via add-on, starting $18/1GB.	[68]

Data on table 4.6 is based on the information on the service provider's websites. Considering storage only cloudnode offers a small amount (100MB), all other providers start with at least 1GB. As the solution is not very demanding, RAM is not a big concern. 512MB or 1GB of RAM offered by the providers should be sufficient. Most providers do not offer information about transfer limitations on their website. The traffic should not be a problem either. Biggest difference in the providers is the pricing. Pricing is either a monthly subscription or price per hour running the server. Nodegear and Modulus do not include database in the base price like the others. Adding a database in Modulus is more expensive than in Nodegear, $1 vs $5. The average specification for a small application host is 512 MB RAM and 1 GB of storage. From the hosts in table 4.6, OpenShift offers 512 MB RAM and 1 GG storage for free. OpenShift is also backed by the redhat organization, which makes it very reliable. Nodegear is an interesting service with cheap fixed monthly price and unlimited storage, but it is not an option due to obscurities discussed in chapter 4.2.1.

4.2.1 Nodegear

Setting up an account for Nodegear brings up a privacy error on chrome:

> "This server could not prove that it is nodegear.io; its security certificate is from lab.castawaylabs.com. This may be caused by a misconfiguration or an attacker intercepting your connection."

Following the Castaway Labs link (http://www.castawaylabs.com/) on the Nodegear website, brings up a setup view of nginx http server. Either the website does not exist or it is misconfigured. Typing http://castawaylabs.com/ directly to browsers URL bar brings up a website of three developers. The website is build with WordPress and half of the text is made of lorem ipsum, the dummy text for making website layouts [74]. Following a link to helpdesk (https://nodegear.zendesk.com/hc/en-us) brings a note that the zendesk account is not active. Because of these problems Nodegear, while cheap, does not seem like a reliable hosting service. Budget is the main requirement for the hosting service, but Nodegear is not reliable enough.

4.2.2 OpenShift

The OpenShift hosting is provided as gears. "Gears are secure containers for your code. Each gear is allocated CPU, memory, disk, and network bandwidth." [106]. The OpenShift [104] free plan includes 3 small gears. A small gear has one CPU, 512MB RAM and 1GB storage [106]. A free account can be created without credit card details. The free plan however includes application idling [105]. The free plan gears are idled due to inactivity. When idle the application can not handle requests and it has to be returned from idle to active state before it can handle requests. When a user tries to use an idled application, it will take some time for the application to move to active state and the user may see a 503 service temporarily unavailable error before the application responds. OpenShift also offers a bronze plan [105] which does not have a monthly fee, includes 3 free small gears and does not idle applications. With the bronze plan the customer can get more computing resources by an hourly fee. The hourly fees start from 0.02€/hour.

An OpenShift deploy template for Pencilblue has been made available [112]. The concern with the openshift free and bronze plans is that free small gears are only available in the US east amazon web services (AWS) server farm. The distance between the servers and the users (US east - Finland) may affect the usability of the service (the latency). A Small.highcpu gear can be deployed in the EU server farm, but they cost 0.025€/hour (219€/year), which is over the budget.

4.2.3 Selected hosting option

Private server is not a competitive option compared to IaaS -providers and dedicated web hosting when price and maintenance are priorities. Dedicated web hosting has far less maintenance work than IaaS -platforms. The IaaS -platforms are roughly the same price as the cheapest dedicated web hosts. Some dedicated web hosting platforms also offer a free plan. Taking advantage of such free plan the solution can be hosted for free and also the amount of maintenance is kept to the minimum.

Nodegear and OpenShift were the most promising of the hosting services in table 4.6. Nodegear is the cheaper option, when OpenShift free plan is not considered. Nodegear is not selected because of the reliablitiy concerns discussed in chapter 4.2.1. Hosting with OpenShift for free the application only be deployed in the US. If the application would be hosted in an EU server with OpenShift, the price would bee

too high. OpenShift free hosting is selected because of the price and because there is already easy way to deploy a pencilblue application. If there will be problems with, for example, latency, then another solution for the hosting will be considered.

5. IMPLEMENTATION

This chapter describes the implementation of the solution.

5.1 Solution overview

The solution is divided into two parts: The frontend and the backend. The solution overview is presented in picture 5.1. Backend is the server that hosts the application and data. Frontend is the client that is using the application. The server and client communicate through the internet using the hypertext transfer protocol (HTTP). The data transferred is hypertext markup language (HTML), javascript, CSS and media files.

The server hardware and the internet connection is provided by the hosting service. The server runs the Nodejs javascript runtime. The application (the CMS), is running on the top of the javascript runtime and handles the requests from the client. The application data is stored in the database on the server.

The client is an internet browser on the users device. The client downloads the application from the server. The application displays the data for the user and makes requests for the server according to the user actions. Additional content is shown from 3rd party services. For example media files from social media services.

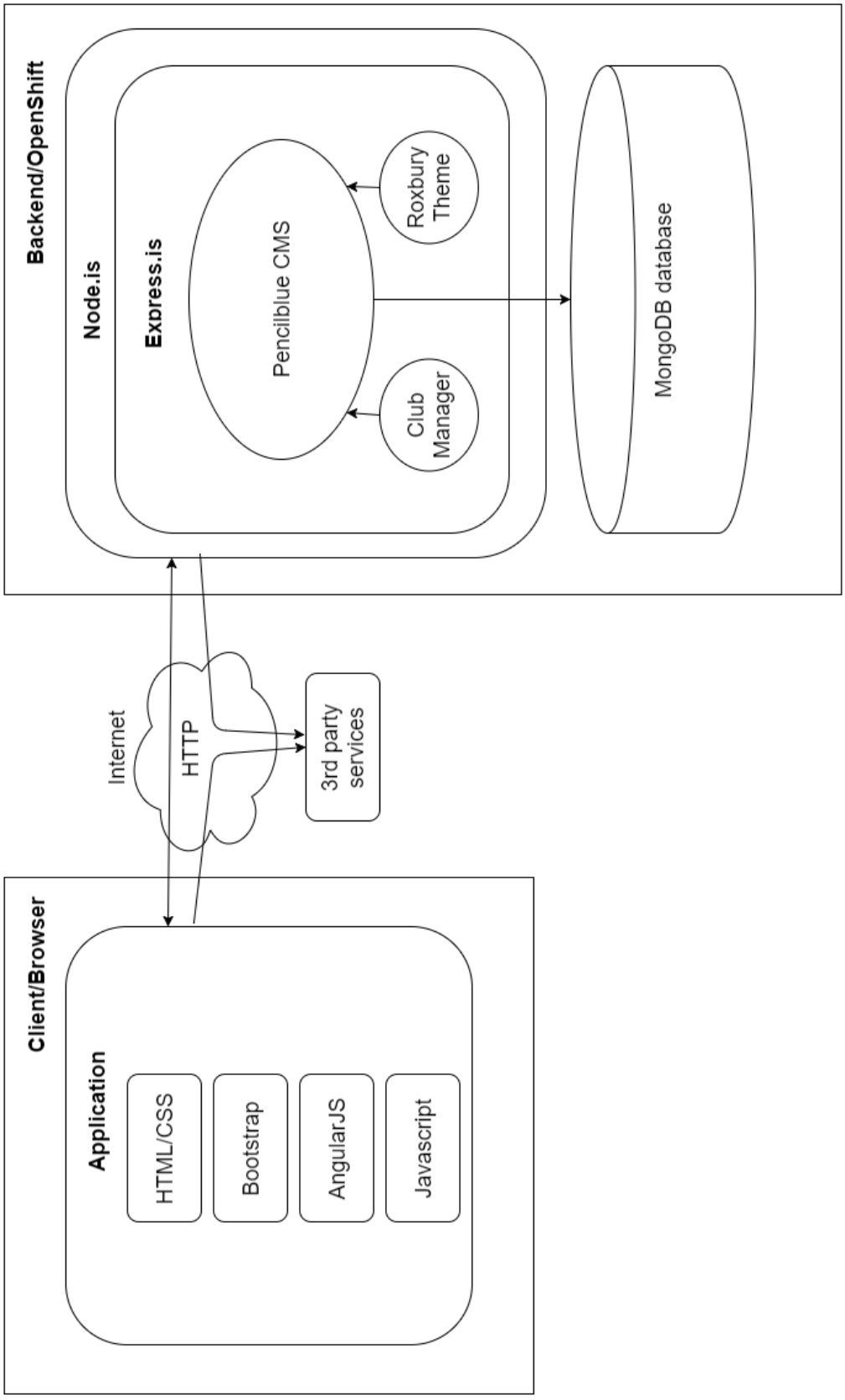

Figure 5.1 *Solution overview*

5.2 Software stack

The system is based on the MEAN stack [95]. MEAN stands for MongoDB database, express application framework, AngularJS javascript framework and Node.js javascript runtime [88]. The software stack is illustrated in figure 5.2

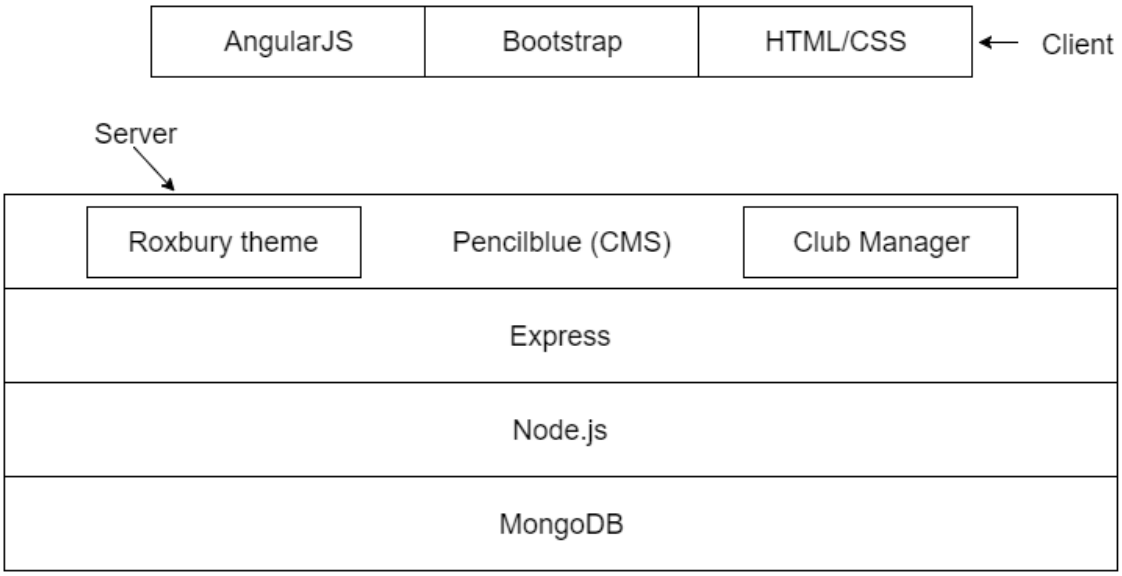

Figure 5.2 *Software stack*

The application data is strored in the MongoDB [98] document database. The application runs on top of the Node.js javascript runtime [103]. The pencilblue CMS application is build using the Express [55] application framework. The club manager plugin extends the pencilblue CMS with club management functions and the Roxbury theme defines the user interface. AngularJS [2] javascript framework is used in the client browser to handle user actions and asynchronous HTTP requests. Bootstrap [17] framework is used to make the user interfaces responsive.

The system file structure is shown in figure 5.3. Pencilblue system and service files are in pencilblue/include folder. The core plugin (pencilblue) is in the plugins folder. Club manager and roxbury theme plugins are also in the plugins folder. The sports club content management is seprated into two plugins so that the data mangament is done in the club manager plugin and the user interface in the roxbury theme. Separating the content management logic and the user interface improves maintenance and makes the user interface changes easier.

The system is hosted in the OpenShift PaaS. The bronze plan is used, because it is free and does not include application idling. The application code is in a git

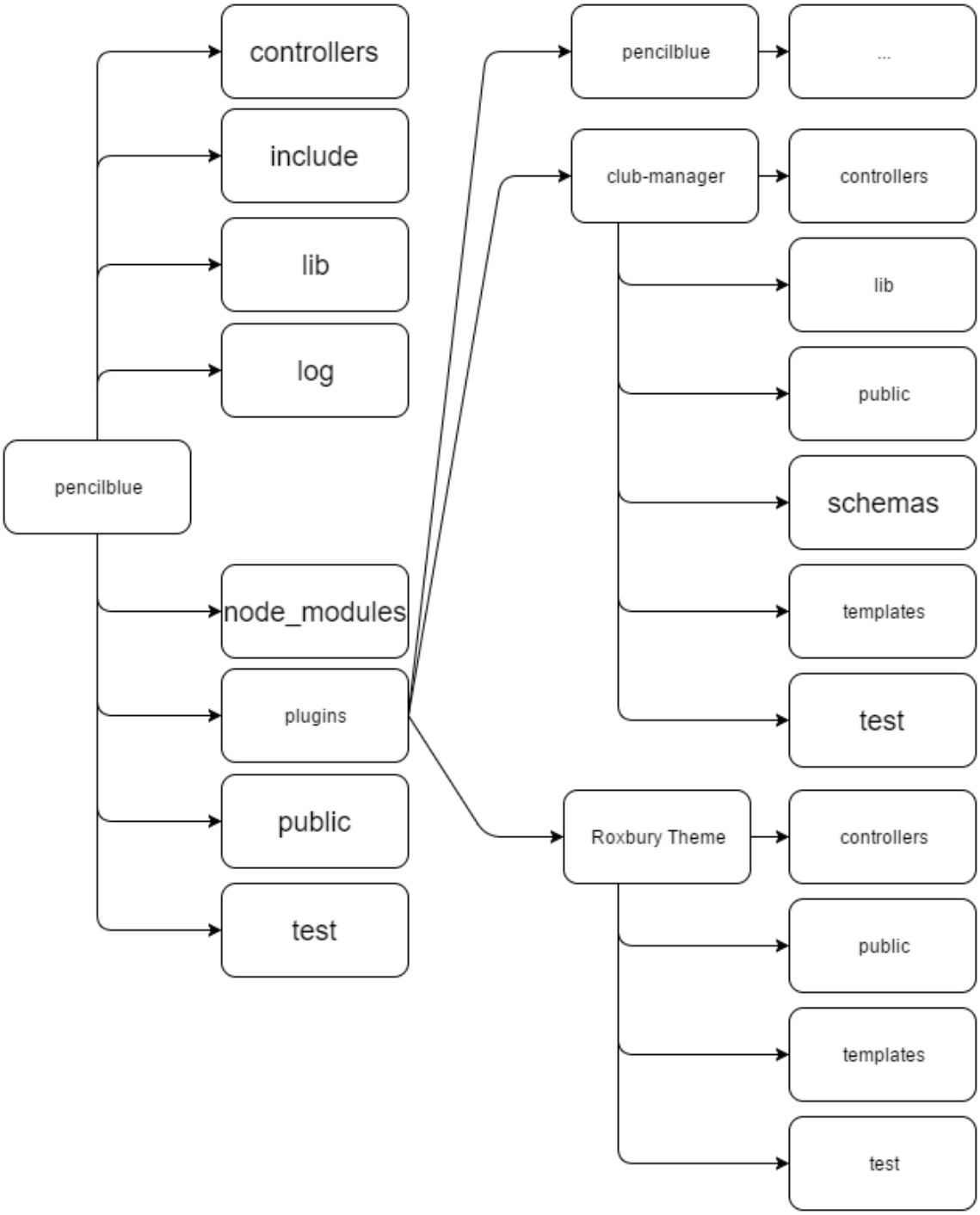

Figure 5.3 *File structure*

repository in the OpenShift server. The application is updated by pushing changes into the repository. When new code is pushed into the OpenShift server repository the application and database are shutdown, the code updated and the application and databse are restarted.

5.3 Pencilblue

The core of the application is the pencilblue CMS. Pencilblue implements all basic features of a content management system. These include creating articles and pages, upoading media files, comment management and user management

Pencilblue defines five user types: reader, writer, editor, managing editor and administrator. The user type defines which features are permitted for the user. Pencilblue provides user registration and user authentication. A new user chooses a username and a password that are used to access the administrator views. Users do not need to register to view the content. Plugins can define the access level for their routes in the controller. It is also possible to define if the route requires authentication. Example is given in the program listing 5.1.

Pencilblue has modular architechture where all content management is done with plugins. The default CMS functionality (post and article management for example) comes from the Pencilblue default plugin called pencilblue (The default plugin has the same name as the platform). Because of the modular architechture, it is easy modify or replace parts of the system with custom functionality. This done by implementing a plugin that replaces the core functionality. A plugin can also implement new functionality. Plugins are added to the plugins-folder and must have a details.js file. Details-file has information about the plugin, including the name, dependencies, settings and permissions that the plugin needs. It also names the main javascript file for the plugin. The main javascript file is used to install, uninstall, load and unload the plugin. Plugins are installed and one theme is selected as the active theme from the pencilblue administrator view.

The architechture follows the Model-View-Controller (MVC) design pattern [45]. Pencilblue MVC architechture is shown in figure 5.4. Pencilblue controllers implement handlers that are mapped to URL patterns. The URL patterns are called routes. When an incoming request matches a route defined in a controller, the handler defined in the controller is invoked. If multiple controllers define handler for

the same route, they are prioritised as follows:

1. Active theme
2. Any plugin
3. Default pencilblue plugin
4. 404 http error is returned if the route does not match any handlers

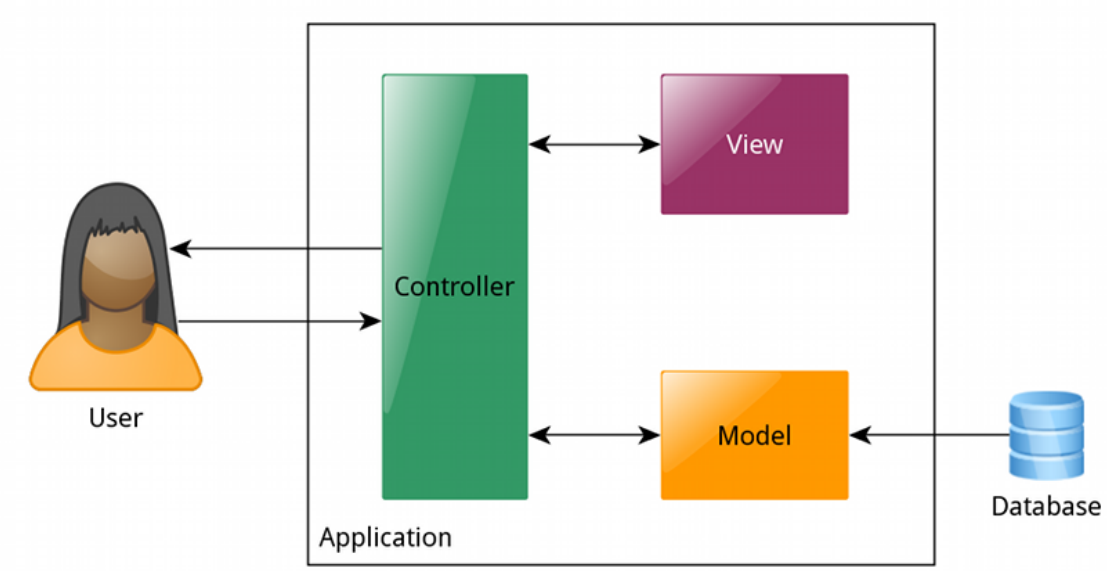

Figure 5.4 *Pencilblue MVC architechture. Image used with permission [69].*

Controllers are defined in the controllers folder. The controllers provide getRoutes-function that the pencilblue platform calls during initialisation. The getRoutes-function returns a list of routes that the controller is responsible for. An example of a getRoutes-function is given in program listing 5.1.

Program 5.1 *Route definition example*

```
ExampleController.getRoutes(cb) {
    var routes = [
        {
            method: 'get',
            path: "/sample",
            auth_required: true,
            access_level: ACCESS_USER,
```

```
            permissions: ["sample_view"],
            content_type: 'text/html'
        }
    ];
    cb(null, routes);
};
```

The controller also defines request handlers for those routes. The default handler is called render and it is called if the route does not specify a handler. There are two types of controllers: Basic controllers and API controllers. Basic controllers do request handling logic and load a template. The template is then returned for the user. API controllers do request handling logic, but data is returned instead of loading and returning a template. API controllers are used to handle asynchronous data requests. Example of a basic controller is given in appendix A and an example of a API controller is given in appendix B. The handlers (saveStats, deleteStats, getStats) in the API controller return data and the render() handler in the basic controller loads and returns the matchStandAlone -template. The basic controller defines a custom pencilblue directive, angular_objects, using the pencilblue template service. Custom directives are used in the templates and provide one way of adding dynamic content into the views. Async.js [94] library is used in the basic controller to manage the asynchronous function calls.

Templates are defined in the templates folder. Templates are HTML files for displaying the data. The template defines how the data is displayed, including display positioning, colors, fonts and user controls. CSS and javascript are used in the HTML template to make the view dynamic and more maintainable. Bootstrap (a HTML, CSS and javascript framework [17]) is used to build the views. The controller can also inject AngularJs components into the template.

In pencilblue the templates are hierarchial and a template can include another template using the template directive defined by a caret (^) notation. For example ^tmp_head^ will ask pencilblue templating engine to load head.html from the templates folder and replace the notation with the loaded template. This allows the reuse of the templates, which speeds development and improves maintainability. If multiple plugins or themes define a template with the same name, one is chosen by the following priority:

1. Active theme
2. Installed plugin
3. Pencilblue core plugin [71]

AngularJs is used to make the templates follow the MVC architechture on the client side. The AngularJs components injected into the template will create the angular controller and the angular scope. The scope is the model of the MCV pattern. Angular controller is created for each page and it is responsible for updating the view according to user actions and other events such as a finished ajax request. In the frontend the view is the HTML file. Template example is given in appendix C. In the listing head, logo, article, team and footer templates are in separate files and pencilblue will subsitute these template directives with the content of those templates. Templates can be loaded deeper in the folder hierarchy using the equal sign (=) in the template directive. For example ^tmp_elements=article^, the article.html template is in templates/elements -folder and in the case of ^tmp_angular=team^, the team.html template is in templates/angular folder. Other included templates are in the templates folder. The rest of the template is HTML, CSS and angularJs directives. The ^angular^ on line 2 is not a template directive. It is a custom directive defined in the controller. An example of the custom directive is given in the basic controller example on appendix A. The pencilblue convention does not force where the models shoud be located or if they are used at all.

Pencilblue database is illustrated in figure 5.5. The figure has a list of pencilblue collections. The custom object type and custom object collections are used to create custom content and are shown in more detail. The custom object type collection defines what data is stored in the custom objects. The possible data fields are [111]:

- text
- number
- wysiwyg - what you see is what you get text editor
- boolean
- date
- peer_object - object reference

- child_objects - list of object references

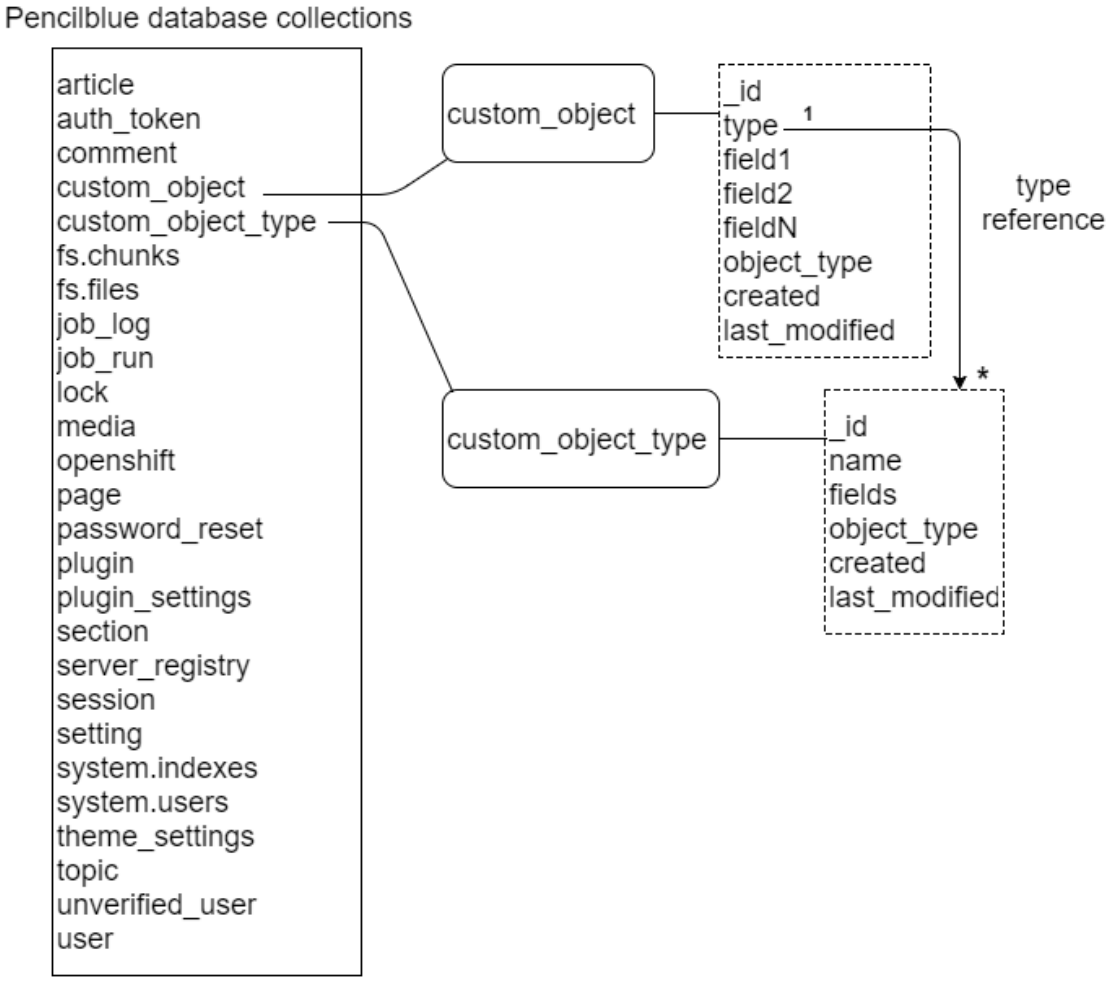

Figure 5.5 *Pencilblue database*

Wysiwyg is field that holds the content that is viewed in a wysiwyg editor. Peer object field is reference to another custom object or a pencilblue object, for example a media object. Child objects field is a list of references to other custom objects or pencilblue objects. The custom object collection stores the actual custom objects that have the data. Each custom object has a type reference to a custom object type. The type is used to find out what fields the custom object document has. Custom objects are managed from the pencilblue administrator views. Example of a custom object administrator view is given in figure 5.6. Example custom object has two text fields and one boolean, date, peer_object, wysiwyg and child_objects field.

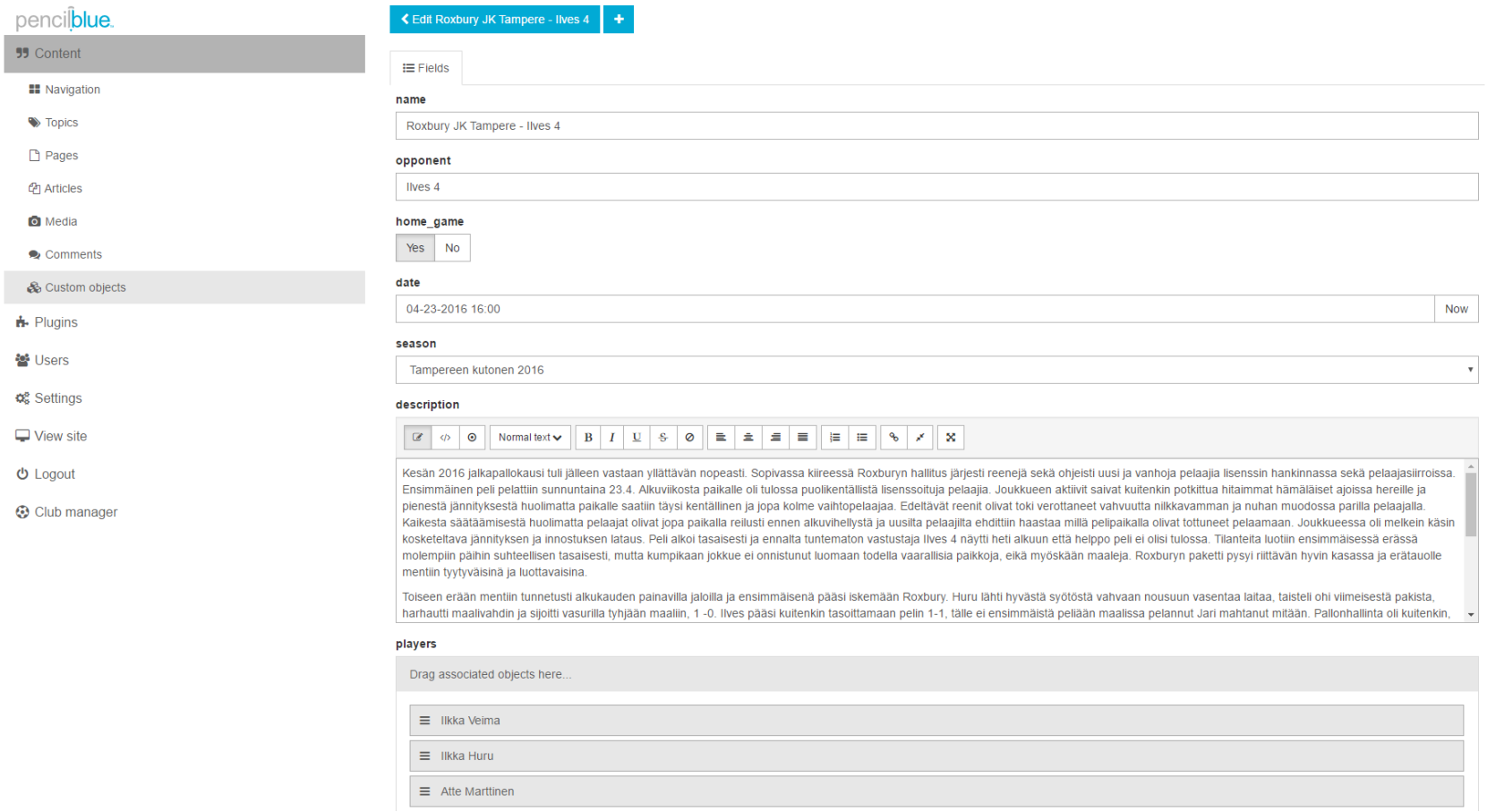

Figure 5.6 *Pencilblue custom object administrator view example*

5.4 Club manager plugin

The club manager plugin extends the pencilblue CMS with the sports club content management. The club manager plugin main javascript file, club-manager.js, defines the functions used to install, uninstall, load and unload the plugin. Installing the club manager plugin creates the club content types (3.2.1) into the database. A new database collection (cm_match_statistics) is created for the player statistics content type. Other content types are stored as pencilblue custom objects. Match statistic have their own collection so that different types of match events can be stored in the same collection. This is possible because a document database (NoSQL) is used instead of an SQL database. For example a goal and an assist have slightly different data, but they can still be stored in the collection. This makes managing the data (save, delete, query) easier. If pencilblue custom objects were used, each event type would need their own custom object type.

The league organizer [108] provides an API for the statistics recored in their system. The statistics are not used because the API gives the statistics as preformatted HTML. It would have been better to receive only the data through the API. Also the league organizer system does not record assists and for these reasons the API is not used.

Club manager database diagram is given in figure 5.7. The figure shows content fields and relations for the custom object types. The figure also shows the custom collection for match statistics. The statistics collection can have three kinds of statistic events: Goal, assist and booking. Each statistic event will have type and match fields, additional fields are shown in the figure for each type of an event.

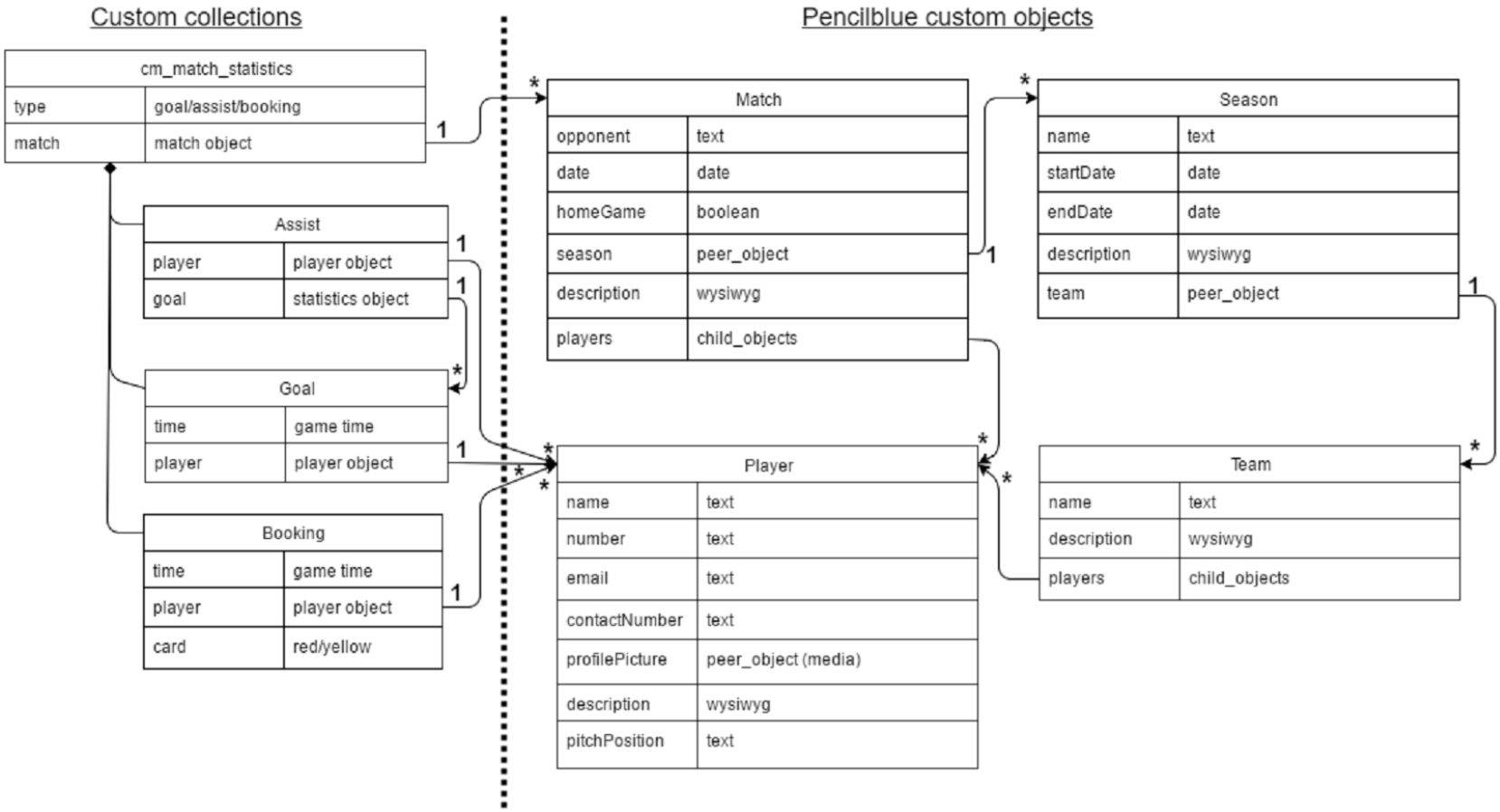

Figure 5.7 *Club manager database*

The controllers are defined in controllers folder, templates in the templates folder and models in a lib folder. The models are used by the controllers to modify the data in the database. The club manager plugin controllers handle request for routes starting with /club-manager/. All club-manager routes are displayed in table 5.1.

Table 5.1 *Club manager routes*

Controller	**Route**
AdminController	/club-manager/admin
MatchApiController	/club-manager/api/stats /club-manager/api/deleteStats /club-manager/api/getstats/
MatchController	/club-manager/match/?name=MatchName
PlayersController	/club-manager/players?team=TeamName&player=playerName
SeasonController	/club-manager/season/?name=SeasonName
TeamController	/club-manager/?team=TeamName

AdminController will return the view for club manager administrator page. The club manager administrator view is used to add statistics to the match reports. Conent types are implemented as pencilblue custom objects are managed through the pencilblue custom object administrator view. The club manager administrator view is shown in figure 5.8. To view the club manager administrator view the user must be logged in. The search box is used to select a match. Then the statistics controls on the right side are used to add or delete goals, assists, and bookings. Game time and player are selected for goals and bookings. Assists are added and linked directly to the goals. All statistics for the match are shown in a list on the right side. MatchApiController handles the API calls (get, save, delete) for the statistics. The statistics are modified asynchronously without page loads.

MatchController handles the requests to view match reports. The match view will display the match report, result and match statistics. URL query parameter name is used to give the name of the match that will be displayed. PlayersController handles requests to view players. The players view will show a list of players and selecting a player will show the details of that player. Details include name, number, image and description. URL query parameter team is used to limit the results to the players of the given team. If team paramter is not given, players for all teams are shown. URL query parameter player can be given to pre select a player. If player parameter is not given the first player of the first team is selected. SeasonController handles requests to view seasons. The season view will show a list of matches in that season. Selecting

a match will show the name, description and statistics for that match. URL query parameter name is used to select which season is shown. TeamController handles requests to view team information. Team view shows a link to players of the team, a list of seasons related to the team and the latest five articles or match reports. URL query parameter team is used select the team. Club manager plugin templates use the default style and navigation of pencilblue. Roxbury theme is used for custom navigation and style. Examples of the non-administrator views when the roxbury theme is in use are given in section 5.5.

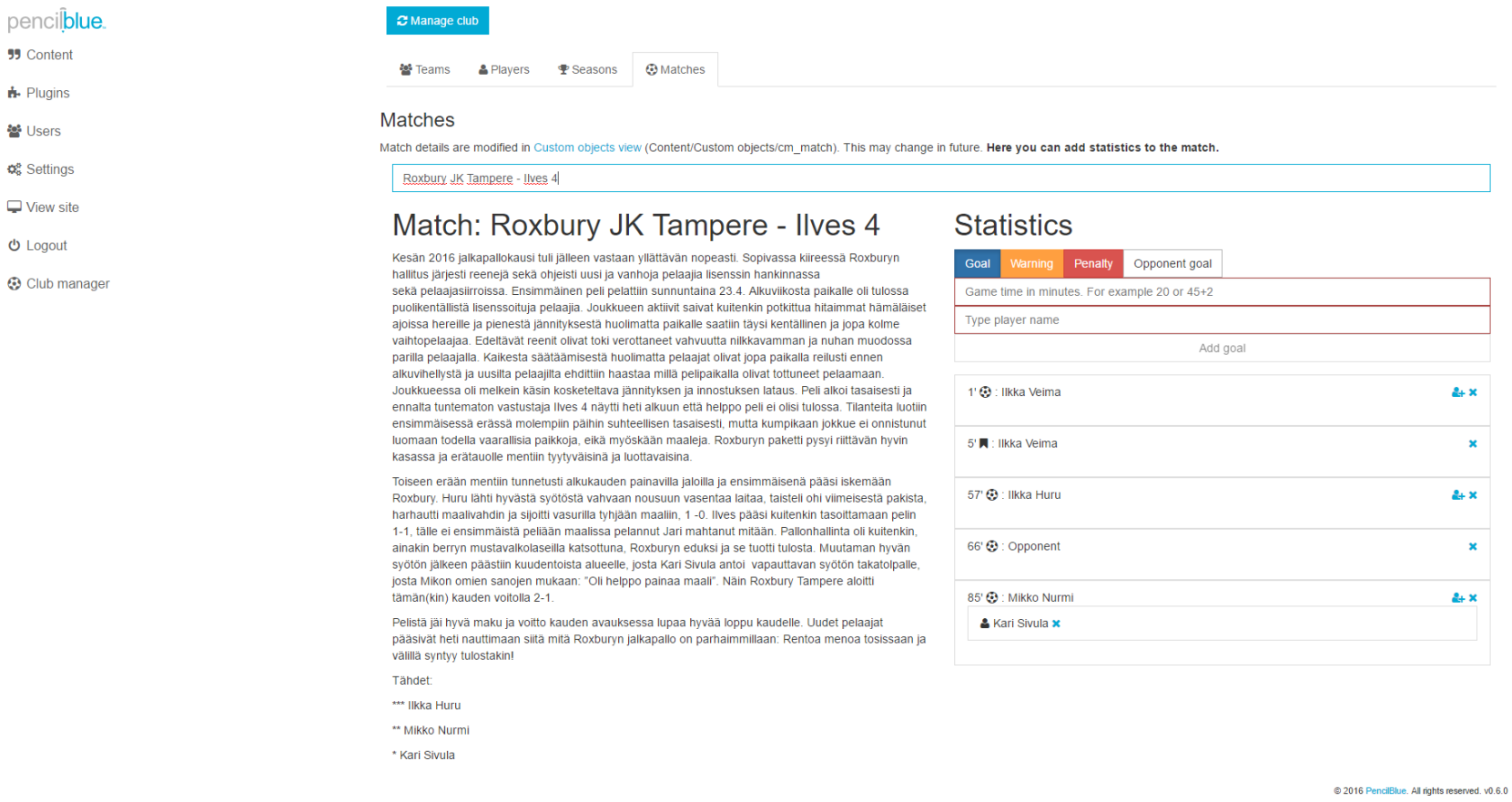

Figure 5.8 *Club manager administrator view*

5.5 Roxbury JK theme

Roxbury JK theme plugin is created to change the user interface to use the color scheme of the Roxbury football club. The color scheme is black, white and greyscale. Font Awesome [58] icons are used to present the statistics: goals (⚽), assists (👤+), yellow cards (🟨) and red cards (🟥). The plugin provides templates for the views required for the sports club content. The templates are placed in the templates folder so that when the theme is set as the active theme, pencilblue will find and use the templates. All templates are build to be responsive so they will be usable on both large and small devices. The responsiveness is achieved using bootstrap [17]. The theme plugin also provides a landing page controller and template. The landing page shows the club logo, a background image, link to about page and links to team pages for both Tampere and Helsinki teams and social media links. The landing page controller overwrites the root route '/'. The logo is a subtemplate that holds the logo image and a link to the landing page. The logo subtemplate is used in all roxbury theme plugin templates. Subtemplates are loaded using the pencilblue template directive described in section 5.3. Figure 5.9 shows the landing page mobile view.

Figure 5.10 shows an example of a match report template. The match report template shows the link to a season and the match name as the topic. Below the topic is the description of the match and in the bottom the result and the match statistics are shown. Statistics also have a links to the players. On the left side is the club logo and a link to the team page. On the right side is a list players that attended the match. The list has a number and name of the player and link to the players profile.

Figure 5.11 shows an example of a players template. Tha player template shows the player image in the middle. Under the image the all time player statistics are shown. The statistics show goals scored, assists, yellow and red cards. On the left side is the logo, a link to the team page and a list players in the team. On the right side is the name, number and the description of the player.

Figure 5.12 shows an example of a season template. The season template shows the selected match description, result and statistics like the match template. The season template and the match template use the same subtemplate to show the match information. On the left side is the logo, a link to the team and a dropdown menu of all the matches in the season. Selecting a match from the dropdown menu

will show the information of that match.

Figure 5.13 shows an example of a team template. The team template shows the latest five match reports or posts for the team. The same subtemplate is used to show the match reports as in the match and season templates. On the left side is the logo, a link to the players in the team and a dropdown menu of all season of the team.

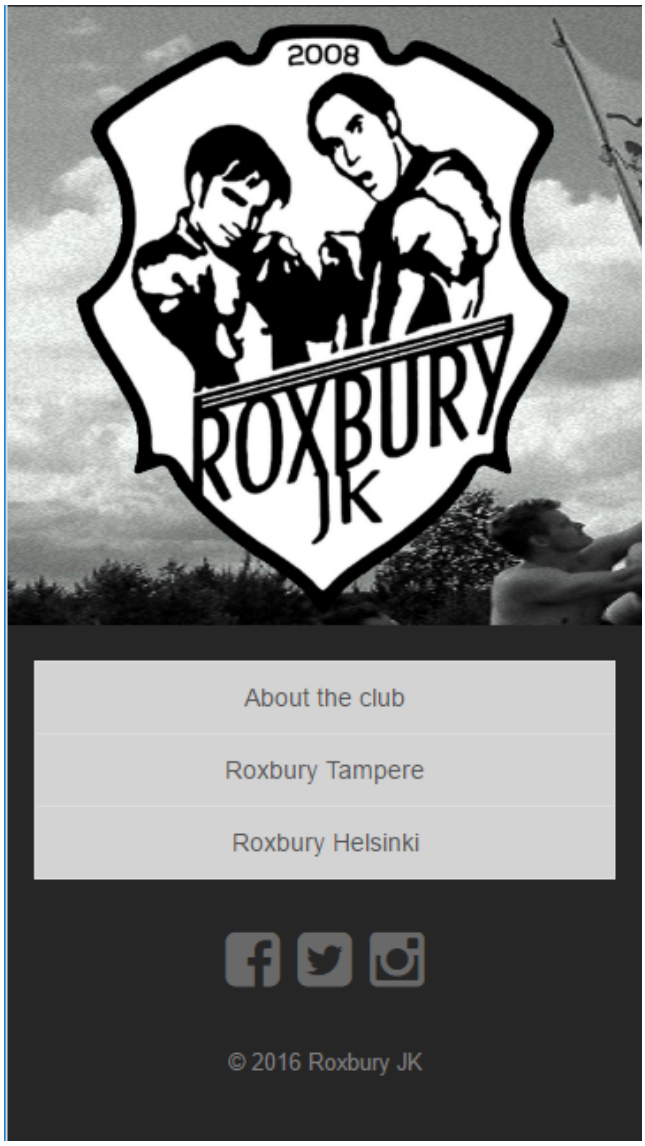

Figure 5.9 *Landing page mobile view*

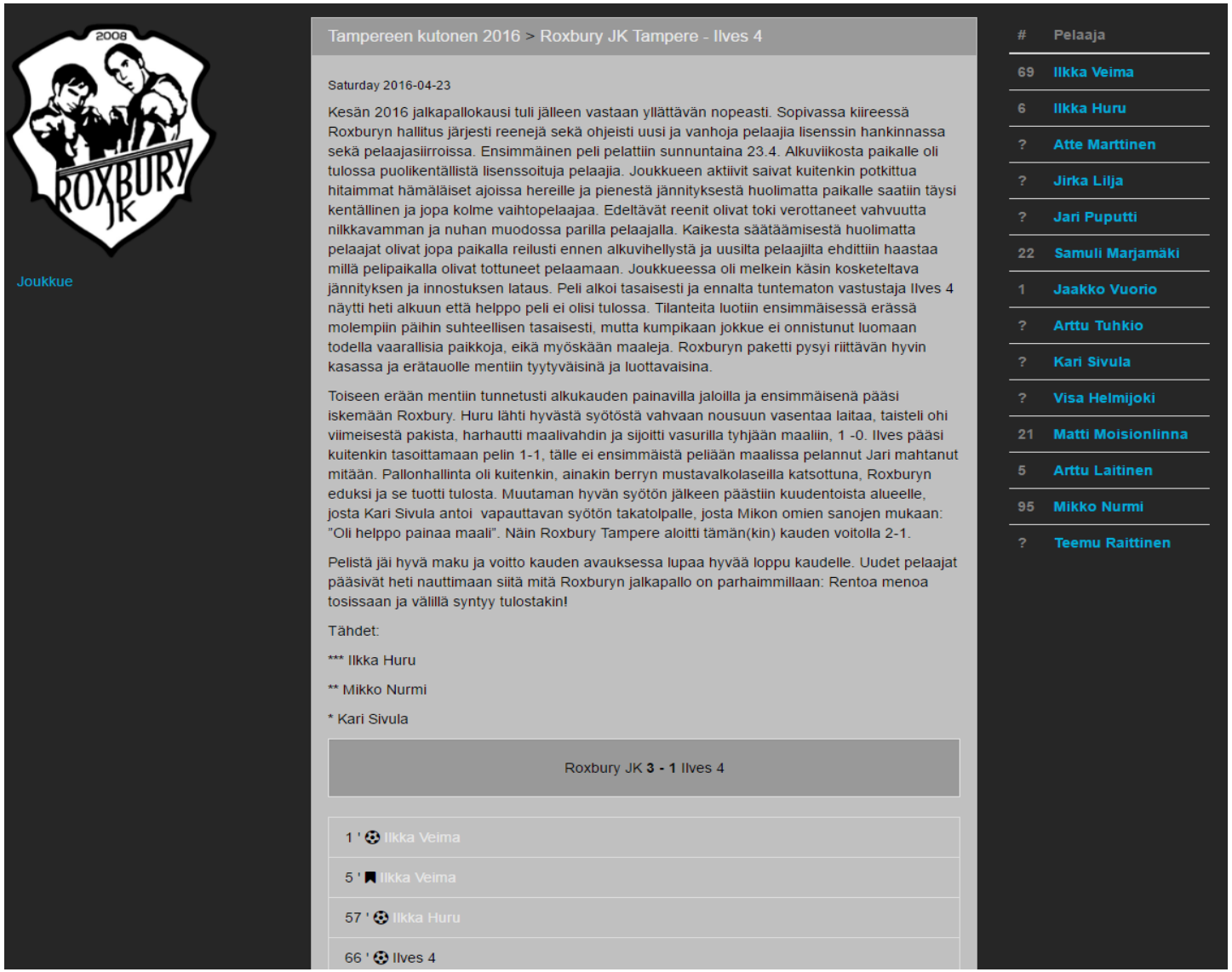

Figure 5.10 *Club manager match view*

Figure 5.11 *Club manager players view*

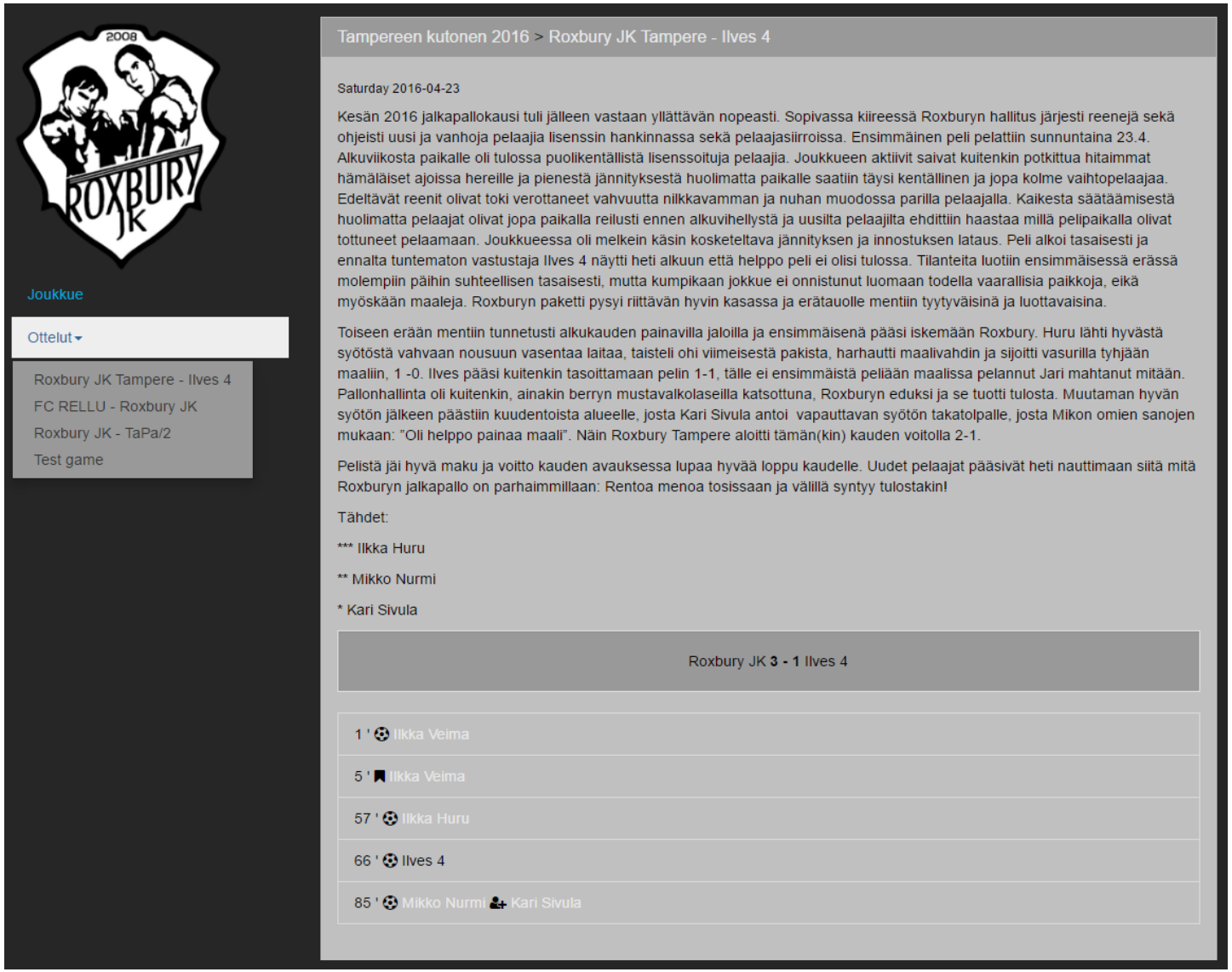

Figure 5.12 *Club manager season view*

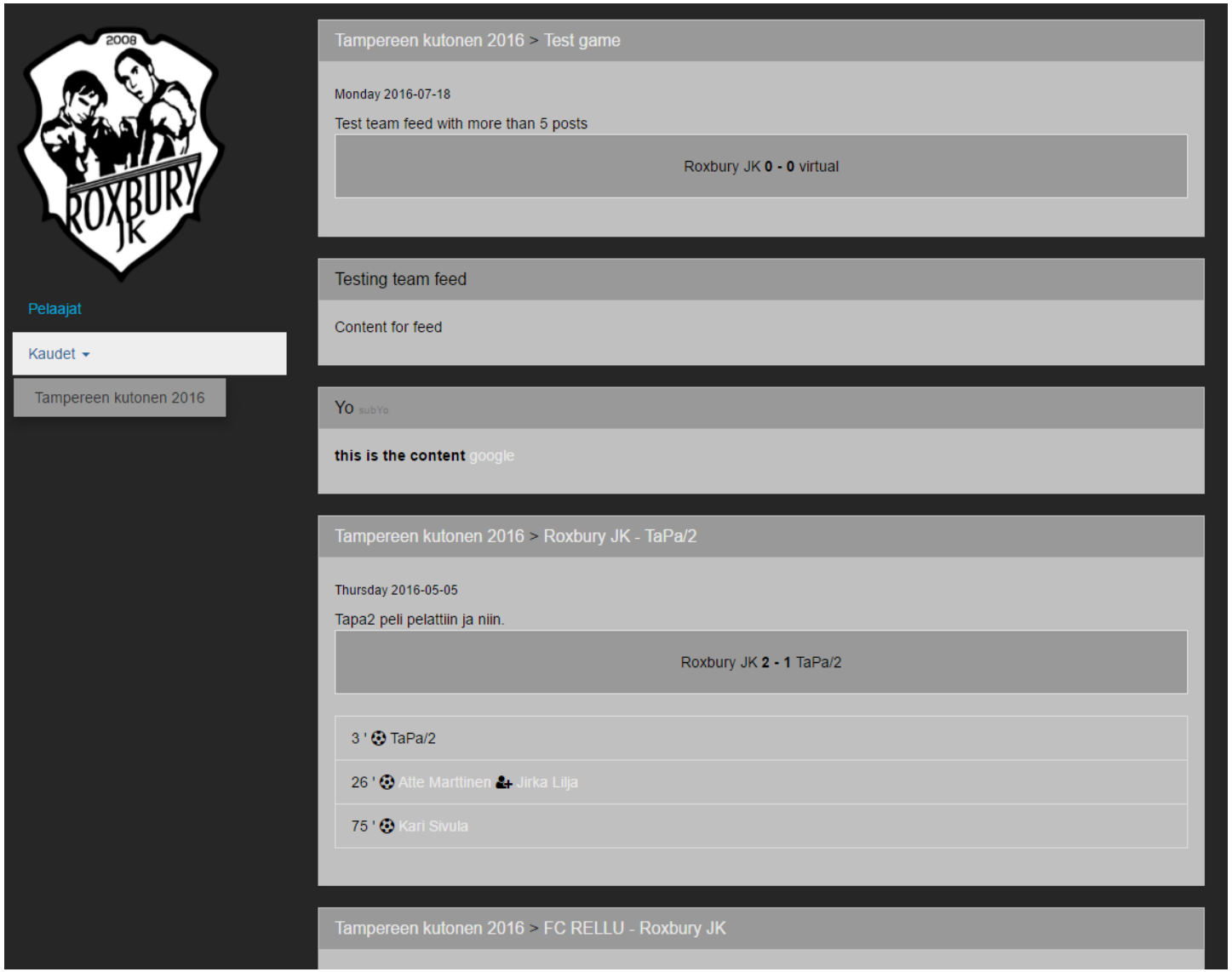

Figure 5.13 *Club manager team view*

5.6 Performance test

A small performance test was run using BlazeMeter [16] free account. Ten virtual users were running an server in EU Central (Frankfurt) AWS server farm. The virtual users were creating HTTP GET requests to following routes:

1. Landing page: http://club-roxburyjk.rhcloud.com/
2. Team page: http://club-roxburyjk.rhcloud.com/club-manager/
   ?team=Roxbury JK Tampere
3. Players page: http://club-roxburyjk.rhcloud.com/club-manager/players
   ?team=Roxbury JK Tampere
4. A match page: http://club-roxburyjk.rhcloud.com/club-manager/season/
   ?name=Tampereen kutonen 2016

The test was run for five minutes and the virtual users would create the requests every 10 seconds. Test summary are shown in figure 5.14. Figures 5.15 and 5.16 show more detailed graphs of response time and latency and it can be seen that the first requests have considerably higher response time than the following requests. The maximum latency and response time is over ten seconds, but the average latency and response time is under 500 milliseconds. From figure 5.17 can be seen that the landing page route is responsible for the high response times for the first requests and is caused by the large banner image on the landing page. As the image is cahced for the following the requests, the response time and latency fall to similar levels as for the other routes. Having an average response time of under 500 milliseconds is considered fast enough.

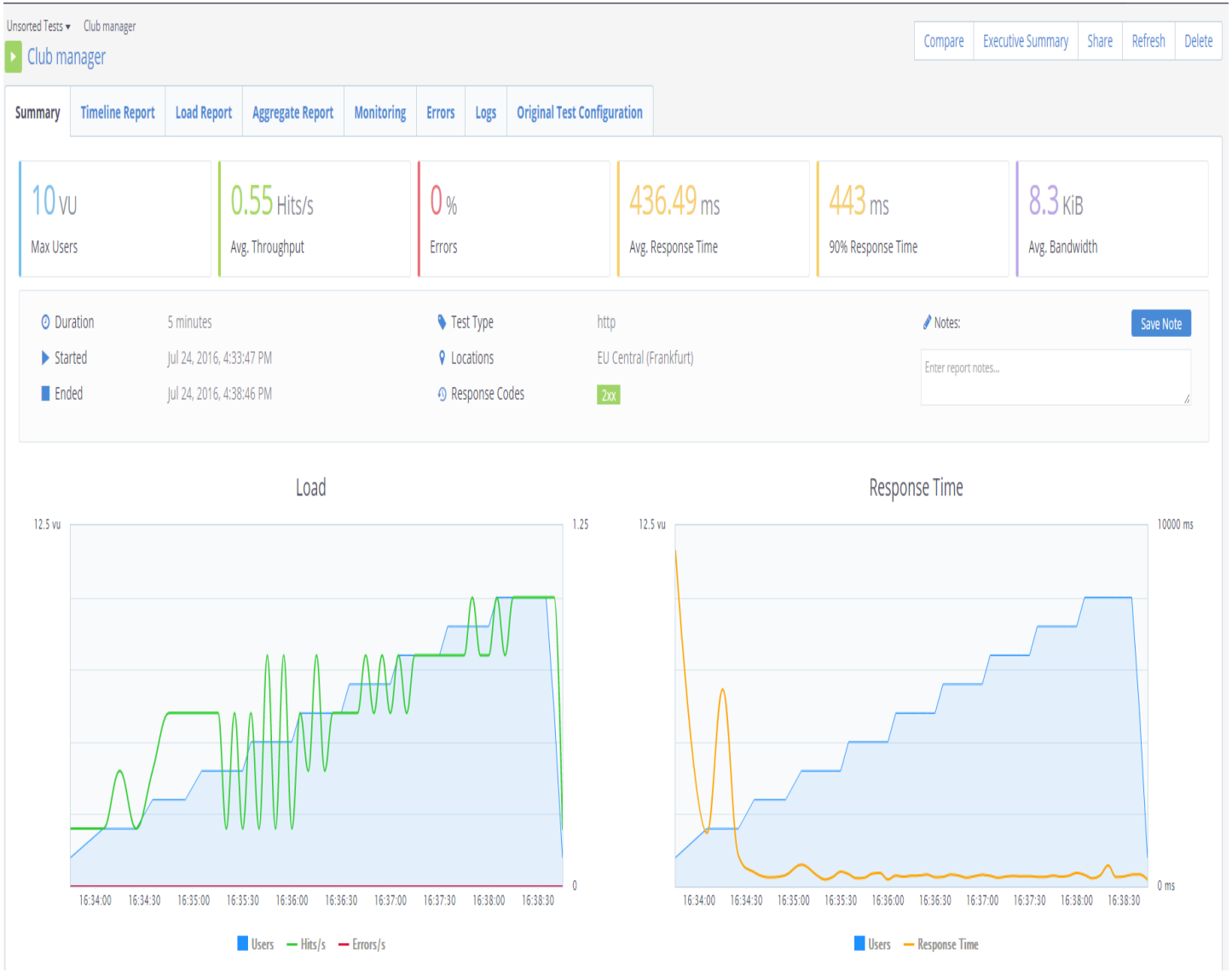

Figure 5.14 *Performance test summary*

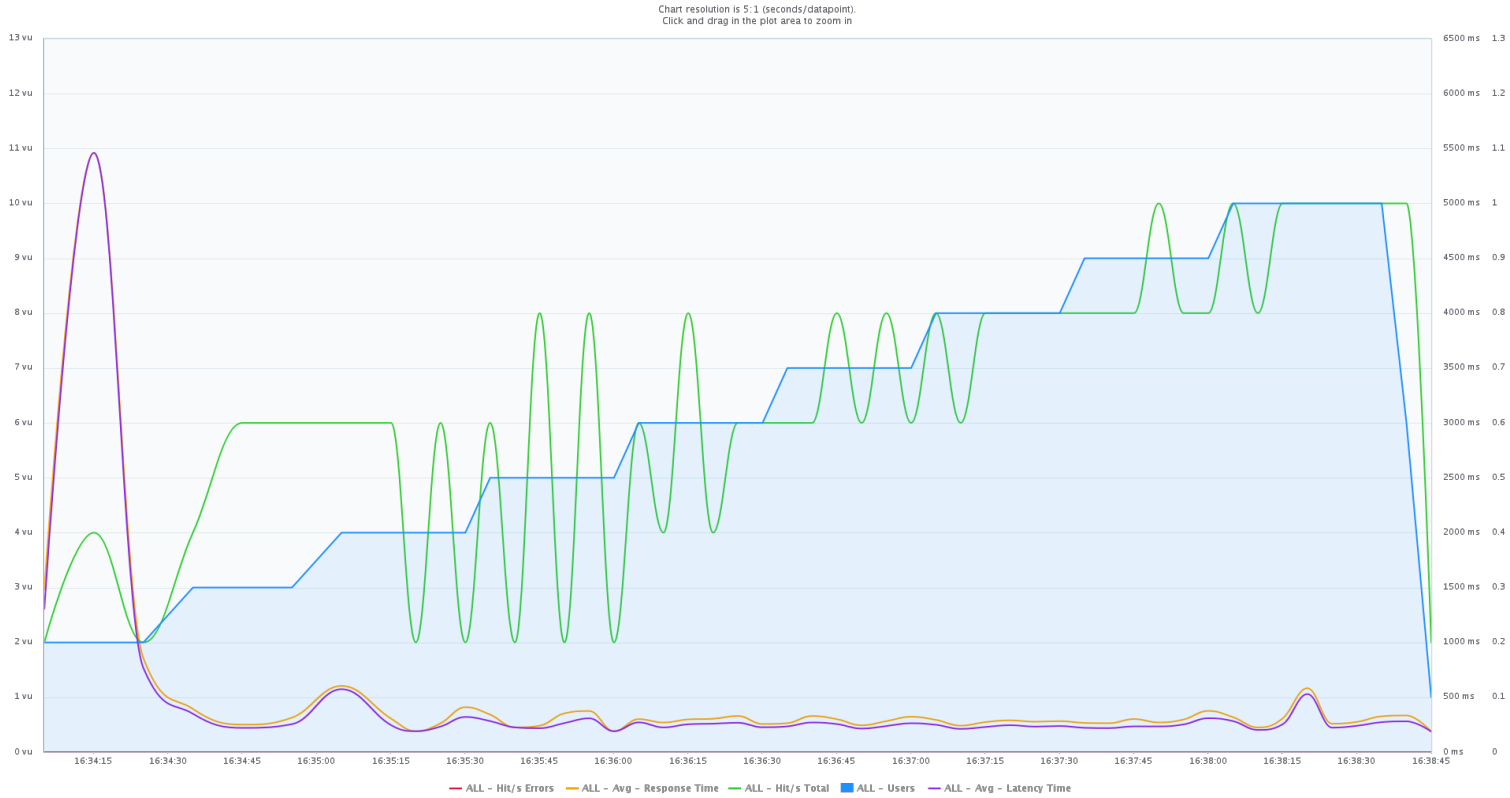

Figure 5.15 *Performance test timeline*

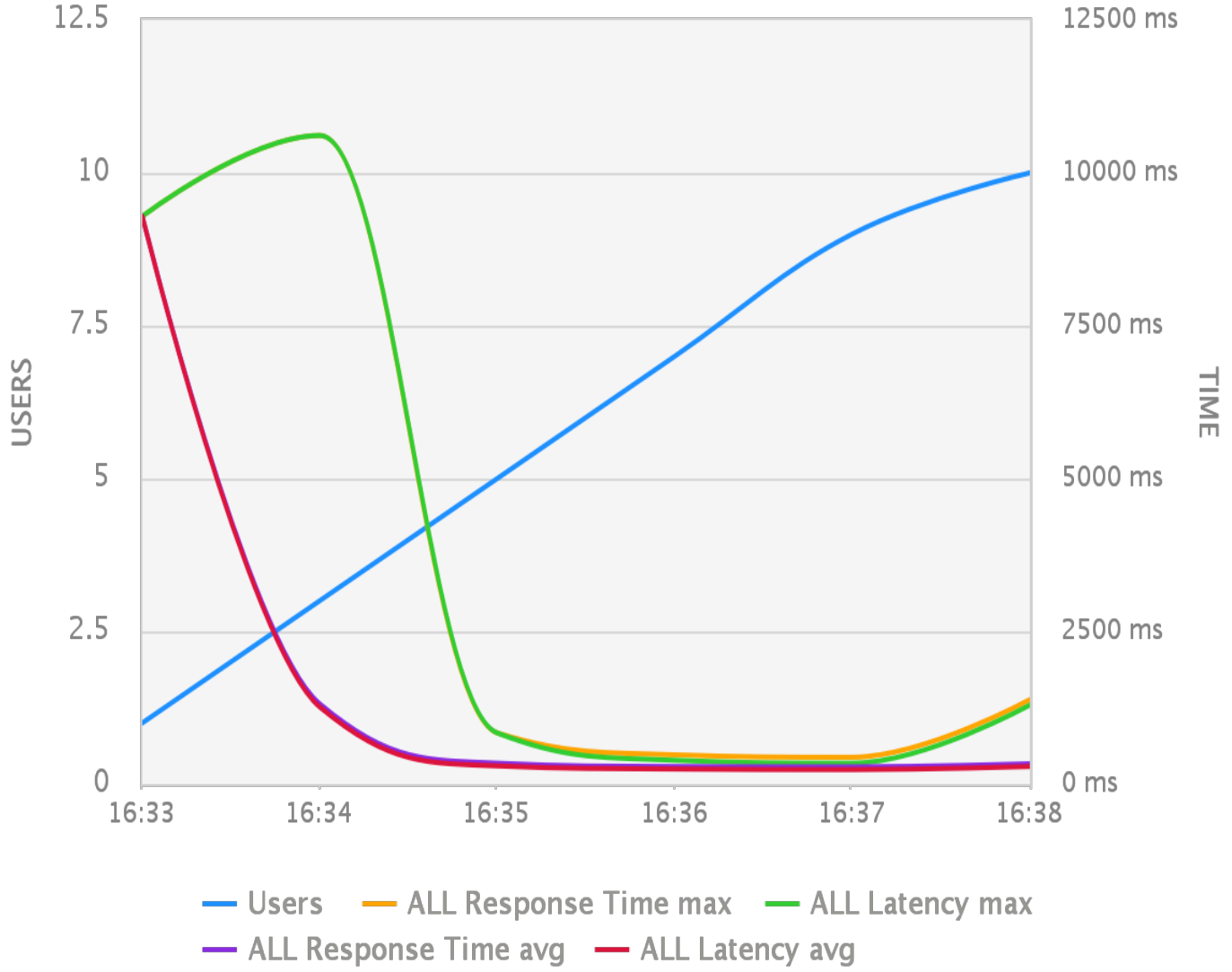

Figure 5.16 *Performance test latency*

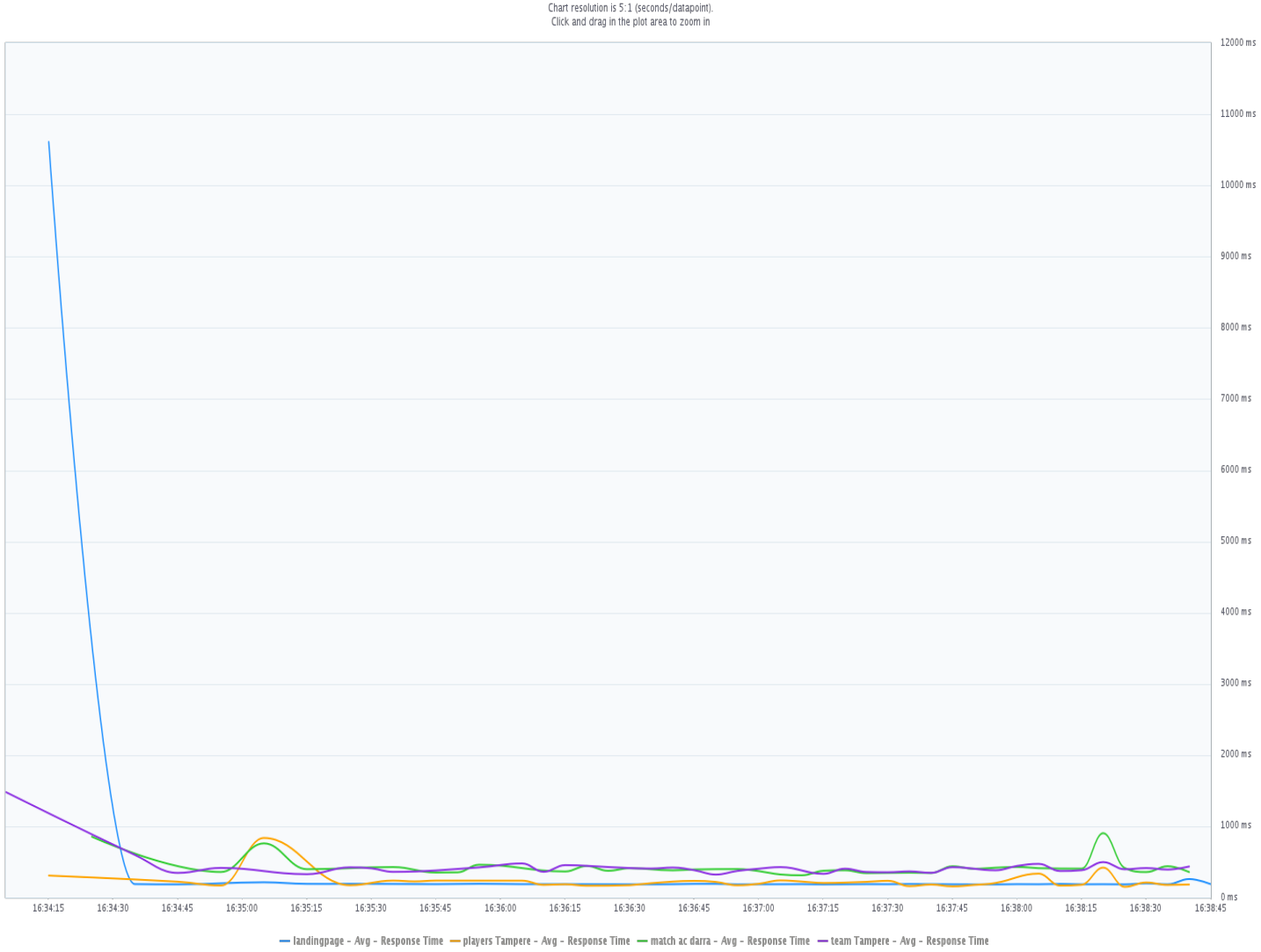

Figure 5.17 *Performance test response times per route*

6. CONCLUSIONS

The environment for content management systems and hosting services is very dynamic. It is difficult to select a platform and a hosting service as new alternatives appear and old alternatives change. Choosing node.js as the server side technology turned out well. Learning one new programming language took a lot of time, so a lot of time was saved when there was no need to learn another one. Also using the node.js based Pencilblue CMS saved a lot of time. Pencilblue is designed to be extendable and the custom object feature made adding new content types easy. The time saved only applies when compared to other custom solution alternatives. If developing a custom system was not an option, then the Joomla with Soccer template or WordPress with either WP Club Manager or SportSpress would probably have been a successful way of building the solution.

6.1 Meeting the requirements

The solution meets the most important requirements. There are five different user types in pencilblue: Reader, writer, editor, managing editor and administrator. These user types satisfy the user requirements described in table 3.1. The solution meets most of the functional requirements described in section 3.2. The readers can consume the generated content. The reader can only view all time statistics for a player. The solution can separate the statistics per season, but the user interfaces do not support this. User interface support is implemented later. Editors can log in to the system and manage the content: articles, pages, teams, players, seasons, matches and match statistics. Content is managed using the pencilblue administrator view for articles, pages and custom objects. Match statistics are managed using the club manager administrator view. Administrators can update the system and the plugins by pushing updates to the openshift server repository.

The solution is build to be responsive and can be accessed using a mobile browser. Both public content and the administrator views are usable in a mobile browser.

The openshift bronze plan offers 1GB storage, which is 10 times more than the current solution. The latency from the server beign in the US is acceptable since it is free to use. The server performance was considered sufficient with tests described in section 5.6.

All content can be managed from the administrator views. In the old system the player information was modified directly using a database client. The solution is almost maintenance free. Database backups are not yet automated, but will be in the future. As the solution is custom build making changes is much easier than with the old system. The solution is also build to be modular which also makes futre changes easier. As all software components are open source and free to use and the hosting is provided for free, it is clear that the solution meets the budget. The solution meets the system requirements. The most important system requirements, the ease of maintenance and budget, are met very well. Not all functional requirements are met, but the development of the solution will continue and the development is made as easy as possible with technology choices.

It is not possible for editors to add sponsor ads on the website. Sponsor ads may be implemented in the future.

6.2 Feedback

The system is not yet deployed to all users, but feedback was asked from one editor. The editor was very pleased with the new solution. The most important thing for the editor was that adding new content was much easier than with the old system. First of all adding new teams was possible and content could be created for each team individually. Adding the content to correct place was also much easier. In the old system when ever a match report or an article was published it had to be manually added to the navigation of the site. This was considered very difficult. In the new system the navigation updates automatically when new content is published. The barrier for creating new content is much lower now than with the old system. The editor would say that the new solution fixes the deficiencies of the old system.

6.3 Criticism

The requirements were not given in much detail and therfore it is difficult to measure how well the solution meets the given requirements. The agile approach to software

development has proved that it is not always best to have detailed requirements. A different way of documenting the requirements and subsecuent changes could have provided a more transparent view into the project.

The technology choices were a bit biased towards a custom solution as one goal of the project was to learn web application development. It would have been possible and likely faster to build the solution using some available service, but a commercial service never meets the requirements as well as a custom solution. Further development will also be faster with the custom solution.

The club manager plugin was build using the default pencilblue theme. This way the implementation became more server oriented. A single page application could have been a better approach. If all club manager controllers were API controllers and only dealt with the data, building a theme would have been more straight forward and the separation of the content management and the user interface would have been more complete.

6.4 Further development

The solution serves the neeeds, but there are chances for improvement and new features can be added. The user interfaces can be improved. The navigation should be unified in the templates. Current navigation is inflexible. For example user can not directly navigate from players page to season page or vise versa. The user has to go through the team page. Also the navigation on the mobile views do not stack currently. It is not a problem yet as there are only few links in the navigation, but later the navigation should be hidden in the mobile views behind a menu button, for example the chrome menu icon (☰).

The club manager administrator view should include all club manager content management and not just the statistics. It is not intuitive way of managing the data from two different administrator views. The custom objects can still be used to store the data, but a management view should be implemented in the club management administrator view, instead of relaying in the pencilblue custom objects administrator view.

A gallery plugin should used to show images on the site. The gallery could show images from the site and the linked social media services. This would give the users

an option to view more content in one place. Social media feeds should be added on the site. For example a facebook plugin that shows the latest posts on the club facebook page.

The landing page controller limits the posts retrived from the server. Other controllers do not limit the data results. It is not a problem as the matches in a season and players in a team are always limited naturally. Really big data queries will not happen. If the descriptions of the matches or players include a lot of media files, then it would be better to implement lazy loading [56]. The data would be loaded only when the user selects to view it. This would improve the response times.

The solution should be wrapped using a container technology like docker [47] so that it becomes easier to maintain and set up new development environments. Using a container could make it easier to change the hosting service. Changing between IaaS-serveices should be almost trivial when using a container technology and at least openshift web hosting offers docker hosting [96]. There are also container as a service (CaaS) providers that host container applications [123][27] [44][73]. The solution should be moved to a server in Europe to reduce the latency. Using a container technology could make the switch easier.

A plugin could be implemented that allows the editors to add sponsor logo or advertisement on the website. This would make attracting sponsors easier.

BIBLIOGRAPHY

[1] aiim.org. (2016) What is web cms (or wcm)? [Online]. Available: http://www.aiim.org/What-is-Web-CMS-WCM-System-Content-Management/

[2] AngularJS. (2016) Angularjs. [Online]. Available: https://angularjs.org/

[3] Anypresence. (2016) The enterprise platform for digital transformation. [Online]. Available: http://www.anypresence.com/

[4] ——. (2016) Why anypresence? [Online]. Available: http://www.anypresence.com/why/anypresence/

[5] apache usergrid_. (2015) Apache usergrid documentation. [Online]. Available: http://usergrid.apache.org/docs/

[6] ——. (2016) The baas framework you run. [Online]. Available: http://usergrid.apache.org/

[7] Apostrophe. (2016) Apostrophe is easy. [Online]. Available: http://apostrophenow.org/

[8] A. T. I. Association. (2016) What is at? [Online]. Available: https://www.atia.org/at-resources/what-is-at/

[9] baasbox. (2016) Baasbox beta. [Online]. Available: http://www.baasbox.com/cloudservice/pricingplan

[10] ——. (2016) Features. [Online]. Available: http://www.baasbox.com/features/

[11] BearDev. (2015) Sport leagues and tournaments management. [Online]. Available: http://extensions.joomla.org/extension/joomsport

[12] ——. (2016) Joomla sport templates. [Online]. Available: http://joomsport.com/joomla-sport-templates.html

[13] ——. (2016) Joomsport module add-ons. [Online]. Available: http://joomsport.com/web-shop/joomsport-addons.html

[14] ——. (2016) Joomsport professional. [Online]. Available: http://joomsport.com/web-shop/professional-version.html

[15] Blake. (2015) Wp: Most popular cms on the planet. [Online]. Available: https://www.balystic.com/blog/wp-most-popular-cms-on-the-planet/

[16] BlazeMeter. (2016) The performance engineering platform for devops. [Online]. Available: https://www.blazemeter.com/

[17] Bootstrap. (2016) Bootstrap is the most popular html, css, and js framework for developing responsive, mobile first projects on the web. [Online]. Available: http://getbootstrap.com/

[18] D. Bosomworth. (2015) Statistics on mobile usage and adoption to inform your mobile marketing strategy. [Online]. Available: http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/

[19] Buckets.io. (2015) Buckets is an open source cms built on node.js and mongodb. [Online]. Available: http://buckets.io/

[20] Budacode. (2015) Node.js cms framework comparison. [Online]. Available: http://blog.budacode.com/2015/05/08/node-js-cms-framework-comparison/

[21] Buildwith. (2015) Cms usage statistics. [Online]. Available: http://trends.builtwith.com/cms

[22] ——. (2016) Framework usage statistics. [Online]. Available: http://trends.builtwith.com/framework

[23] built.io. (2015) Introduction. [Online]. Available: https://docs.built.io/

[24] Built.io. (2016) Built.io for students. [Online]. Available: https://www.built.io/students

[25] ——. (2016) Developer guide. [Online]. Available: https://docs.built.io/guide

[26] B. Callens. (2015) Working with custom objects. [Online]. Available: https://pencilblue.org/article/working-with-custom-objects-20150226

[27] D. cloud. (2016) Build, ship, and run - any app, anywhere. [Online]. Available: https://cloud.docker.com/

[28] Cloudnode. (2016) Your apps and data fully managed. [Online]. Available: https://cloudno.de/

[29] clouds360.com. (2015) The top 20 infrastructure as a service (iaas) vendors. [Online]. Available: http://www.clouds360.com/iaas.php

[30] ——. (2015) The top 20 platform as a service (paas) vendors. [Online]. Available: http://www.clouds360.com/paas.php

[31] ——. (2015) The top 20 software as a service (saas) vendors. [Online]. Available: http://www.clouds360.com/saas.php

[32] Clubpress. (2015) Adds a list of players career appearances to their profile. [Online]. Available: https://wpclubmanager.com/extensions/player-appearances/

[33] ——. (2015) Browse the collection of plugin extensions. [Online]. Available: https://wpclubmanager.com/extensions/

[34] ——. (2015) Display a gallery of players or staff anywhere. [Online]. Available: https://wpclubmanager.com/extensions/players-gallery/

[35] ——. (2015) Display score summaries for matches. [Online]. Available: https://wpclubmanager.com/extensions/score-summary/,

[36] ——. (2015) A responsive, multi-sport theme for wp club manager. [Online]. Available: https://wpclubmanager.com/themes/scoreline/

[37] ——. (2015) Supercharge your club sponsors. [Online]. Available: https://wpclubmanager.com/extensions/sponsors-pro/

[38] ——. (2015) Wp club manager - manage your sports club. [Online]. Available: https://wordpress.org/support/plugin/wp-club-manager

[39] ——. (2016) Wp club manager - manage your sports club. [Online]. Available: https://wordpress.org/plugins/wp-club-manager/

[40] Clupress. (2015) Terms & conditions. [Online]. Available: https://wpclubmanager.com/terms-and-conditions/

[41] T. Creco. (2015) Exploring different cms solutions built on node.js. [Online]. Available: http://www.sitepoint.com/exploring-different-cms-solutions-built-on-node-js/

[42] C. Critic. (2015) Cms list and list of content management systems. [Online]. Available: https://www.cmscritic.com/dir/cms/

[43] deployed. (2016) The simplest way to build an api. [Online]. Available: http://deployd.com/

[44] Deployit. (2016) Apps cloud hosting you don‘t need to manage. [Online]. Available: https://deployit.io/

[45] C. Developer. (2016) Mvc architecture. [Online]. Available: https://developer.chrome.com/apps/app_frameworks

[46] DigitalOcean. (2016) Simple pricing. [Online]. Available: https://www.digitalocean.com/pricing/

[47] Docker. (2016) Build, ship, run. [Online]. Available: https://www.docker.com/

[48] Drupal.org. (2016) About. [Online]. Available: https://www.drupal.org/about

[49] ——. (2016) Drupal sports & league management. [Online]. Available: https://groups.drupal.org/sports-club-league-management

[50] ——. (2016) Usage statistics for league: Online tournament manager. [Online]. Available: https://www.drupal.org/project/usage/league

[51] ——. (2016) Usage statistics for sports club management. [Online]. Available: https://www.drupal.org/project/usage/sport

[52] drupalthemesamples. (2016) League: Online tournament manager. [Online]. Available: https://drupalthemesamples.com/league/

[53] T. E. Erian. (2014) The xaas family: Understanding iaas, paas and saas. [Online]. Available: http://archive.thoughtsoncloud.com/2014/10/xaas-family-iaas-paas-saas-explained/

[54] evennode. (2016) Node.js and io.js hosting for your apps. [Online]. Available: http://www.evennode.com/

[55] Express. (2016) Fast, unopinionated, minimalist web framework for node.js. [Online]. Available: https://expressjs.com/

[56] M. Fowler. (2016) Lazy load. [Online]. Available: http://martinfowler.com/eaaCatalog/lazyLoad.html

[57] P. Froberg. (2016) What is freemium ? [Online]. Available: http://www.freemium.org/what-is-freemium-2/

[58] D. Gandy. (2016) Font-awesome the iconic font and css toolkit. [Online]. Available: https://fortawesome.github.io/Font-Awesome/

[59] github. (2016) Baasbox: Contributions to master, excluding merge commits. [Online]. Available: https://github.com/baasbox/baasbox/graphs/contributors

[60] ——. (2016) Calipso: Contributions to master, excluding merge commits. [Online]. Available: https://github.com/cliftonc/calipso/graphs/contributors

[61] ——. (2016) Deployed: Contributions to master, excluding merge commits. [Online]. Available: https://github.com/deployd/deployd/graphs/contributors

[62] ——. (2016) Keystone: Contributions to master, excluding merge commits. [Online]. Available: https://github.com/keystonejs/keystone/graphs/contributors

[63] ——. (2016) Pencilblue: Contributions to master, excluding merge commits. [Online]. Available: https://github.com/pencilblue/pencilblue/graphs/contributors

[64] ——. (2016) Usergrid: Contributions to master, excluding merge commits. [Online]. Available: https://github.com/apache/usergrid/graphs/contributors?from=2011-09-25&to=2016-03-02&type=c

[65] M. Giza. (2013) Cloud mbaas and open source mbaas: What's the difference? [Online]. Available: http://searchsoa.techtarget.com/feature/Cloud-MBaaS-and-open-source-MBaaS-Whats-the-difference

[66] D. Group. (2014) Df group. [Online]. Available: https://www.youtube.com/channel/UCB4Jqz_oq5qxEGujBMqQe0g

[67] D. Hellmann. (2013) Dynamic code patterns: Extending your applications with plugins. [Online]. Available: http://docs.openstack.org/developer/stevedore/essays/pycon2013.html

[68] heroku.com. (2016) Simple, flexible pricing. [Online]. Available: https://www.heroku.com/pricing

[69] B. Hyder. (2014) An introduction to pencilblue controllers. [Online]. Available: https://pencilblue.org/article/introduction_to_controllers

[70] ——. (2015) Quickstart: Plugins. [Online]. Available: https://github.com/pencilblue/pencilblue/wiki/Quickstart:-Plugins

[71] ——. (2015) Quickstart: Html templates. [Online]. Available: https://github.com/pencilblue/pencilblue/wiki/Quickstart:-HTML-Templates

[72] A. Incubator. (2015) Apache usergrid project incubation status. [Online]. Available: http://incubator.apache.org/projects/usergrid.html

[73] P. internet. (2016) Docker pilvipalveluna tehty kehittÃ¤jille. palvelimet suomessa. [Online]. Available: https://www.planeetta.net/docker/

[74] ipsum.com. (2016) Lorem ipsum. [Online]. Available: http://www.lipsum.com/

[75] M. Johnston. (2010) What is a cms? [Online]. Available: http://www.cmscritic.com/what-is-a-cms/

[76] ——. (2011) Cms or wcm - which is which? [Online]. Available: http://www.cmscritic.com/cms-or-wcm-which-is-which/

[77] Joomla. (2016) Core features. [Online]. Available: https://www.joomla.org/core-features.html

[78] ——. (2016) Joomla! [Online]. Available: https://www.joomla.org/

[79] Joomsharper. (2015) Sp soccer - sport component. [Online]. Available: https://www.joomshaper.com/joomla-extensions/sp-soccer

[80] ——. (2016) Sport team clubs joomla template. [Online]. Available: https://www.joomshaper.com/joomla-templates/soccer

[81] ——. (2016) Terms of use. [Online]. Available: https://www.joomshaper.com/terms-of-use

[82] M. Kavis. (2016) Don't let vendor lock-in fears devalue your cloud initiative. [Online]. Available: http://www.forbes.com/sites/mikekavis/2016/01/21/dont-let-vendor-lock-in-fears-devalue-your-cloud-initiative/2/#2dc2942179df

[83] Kii. (2016) Iot platform for smart apps. [Online]. Available: https://en.kii.com/

[84] K. Lane. (2015) Backend as a service (baas). [Online]. Available: http://baas.apievangelist.com/

[85] LeagueRepublic.

[86] ——. (2015) Club website. [Online]. Available: https://www.leaguerepublic.com/features/club-website.html

[87] ——. (2015) Simple pricing. [Online]. Available: https://www.leaguerepublic.com/pricing.html

[88] Linnovate. (2016) The friendly & fun javascript fullstack for your next web application. [Online]. Available: http://mean.io/#!/

[89] A. Loponen, "Web-sisällönhallintajärjestelmien ominaisuudet," *TamPub julkaisuarkisto*, p. 8, May 2012.

[90] ——, "Web-sisällönhallintajärjestelmien ominaisuudet," *TamPub julkaisuarkisto*, p. 14, May 2012.

[91] ——, "Web-sisällönhallintajärjestelmien ominaisuudet," *TamPub julkaisuarkisto*, pp. 30–37, May 2012.

[92] ——, "Web-sisällönhallintajärjestelmien ominaisuudet," *TamPub julkaisuarkisto*, p. 18, May 2012.

[93] T. Lütke. (2014) Mobile now accounts for 50.3% of all ecommerce traffic. [Online]. Available: https://www.shopify.com/blog/15206517-mobile-now-accounts-for-50-3-of-all-ecommerce-traffic/

[94] C. McMahon. (2016) Async. [Online]. Available: http://caolan.github.io/async/

[95] MEAN.IO. (2016) The friendly & fun javascript fullstack for your next web application. [Online]. Available: http://mean.io/#!/

[96] C. Milsted. (2016) Getting any docker image running in your own openshift cluster. [Online]. Available: https://blog.openshift.com/getting-any-docker-image-running-in-your-own-openshift-cluster/

[97] Modulus. (2016) Deploy, scale, and monitor apps in the language of your choice. [Online]. Available: https://modulus.io/

[98] MongoDB. (2016) Building on the best of relational with the innovations of nosql. [Online]. Available: https://www.mongodb.com/

[99] M. Monroe. (2013) The gospel of mbaas- according to anypresence co-founder (part 1 of 2). [Online]. Available: http://www.infoq.com/news/2013/05/MBaaS-Anypresence

[100] P. Newenham. (2015) Irish-founded sports technology firm teamer acquires club website. [Online]. Available: http://www.irishtimes.com/business/technology/irish-founded-sports-technology-firm-teamer-acquires-club-website-1.2426320

[101] Nodegear. (2016) Node.js hosting for web developers. [Online]. Available: https://nodegear.com/

[102] Node.js. (2016) About. [Online]. Available: https://nodejs.org/en/about/

[103] ——. (2016) Node.js. [Online]. Available: https://nodejs.org/en/

[104] OpenShift. (2016) Develop, host, and scale your apps in the cloud. [Online]. Available: https://www.openshift.com/

[105] ——. (2016) Monthly plan comparison. [Online]. Available: https://www.openshift.com/pricing/plan-comparison.html

[106] ——. (2016) Plans & pricing. [Online]. Available: https://www.openshift.com/pricing/index.html

[107] Osallistujat.com. (2016) osallistujat.com. [Online]. Available: http://www.osallistujat.com/

[108] Palloliitto. (2015) Uudistettu taso-ohjelma avattu joukkueiden käyttöön. [Online]. Available: https://www.palloliitto.fi/uutiset/uudenmaan-piiri/uudistettu-taso-ohjelma-avattu-joukkueiden-kayttoon/

[109] ——. (2016) Pelipaikka. [Online]. Available: https://www.palloliitto.fi/pelipaikka

[110] R. Palmeri. (2016) Why facebook's parse shutdown is good news for all of us. [Online]. Available: http://venturebeat.com/2016/01/30/why-facebooks-parse-shutdown-is-good-news-for-all-of-us/

[111] Pencilblue. (2016) Custom object service. [Online]. Available: http://pencilblue.github.io/files/include_service_entities_custom_object_service.js.html#l105

[112] ——. (2016) The full featured node.js cms you've been waiting for. [Online]. Available: https://hub.openshift.com/quickstarts/159-pencilblue

[113] Pitchero. (2015) How can i have my own custom design? [Online]. Available: http://help.pitchero.com/knowledgebase/articles/619401-how-can-i-have-my-own-custom-design

[114] ——. (2015) Pitchero. [Online]. Available: http://www.pitchero.com/

[115] L. I. Project. (2006) Vendor lock-in definition. [Online]. Available: http://www.linfo.org/vendor_lockin.html

[116] E. Protalinski. (2015) Wordpress now powers 25% of the web. [Online]. Available: http://venturebeat.com/2015/11/08/wordpress-now-powers-25-of-the-web/

[117] A. Rabbetts. (2013) Cloud supplier lock-in - our experience. [Online]. Available: http://www.computerweekly.com/opinion/Cloud-vendor-lock-in-our-experience

[118] M. Rouse. (2012) vendor lock-in. [Online]. Available: http://searchcloudapplications.techtarget.com/definition/vendor-lock-in

[119] Sencha. (2016) Sencha touch. [Online]. Available: https://www.sencha.com/products/touch/#overview

[120] S. C. M. Services. (2015) Sports club management services. [Online]. Available: http://www.sportsclubservices.com/demo1.cfm

[121] ——. (2015) Sports club management services. [Online]. Available: http://www.sportsclubservices.com/fees.cfm

[122] A. Siles. (2011) League: Online tournament manager. [Online]. Available: https://www.drupal.org/project/league

[123] Sloppy.io. (2016) Docker hosting cloud container and microservices the devops way. [Online]. Available: https://sloppy.io/

[124] S. Sridhar. (2012) Mobile cloud backend as a service ecosystem map - all roads lead to baas. [Online]. Available: http://www.kinvey.com/blog/65/mobile-cloud-backend-as-a-service-ecosystem-map-8211-all-roads-lead-to-baas

[125] E. Staff. (2015) Wordpress vs joomla vs drupal - which one is better? [Online]. Available: http://www.wpbeginner.com/opinion/wordpress-vs-joomla-vs-drupal-which-one-is-better/

[126] R. support. (2016) Cms comparison: Drupal, joomla, and wordpress. [Online]. Available: https://www.rackspace.com/knowledge_center/article/cms-comparison-drupal-joomla-and-wordpress

[127] teamer. (2015) Features. [Online]. Available: https://teamer.net/features

[128] ——. (2015) teamer payments. [Online]. Available: https://teamer.net/pages/payments

[129] Teamsnap. (2015) Member availability. [Online]. Available: https://www.teamsnap.com/teams/features/member-availability

[130] ——. (2015) Messages. [Online]. Available: https://www.teamsnap.com/teams/features/messages

[131] ——. (2015) Payments. [Online]. Available: https://www.teamsnap.com/teams/features/payments

[132] ——. (2015) Plans and pricing. [Online]. Available: https://www.teamsnap.com/teams/plans-and-pricing

[133] ——. (2015) Rosters. [Online]. Available: https://www.teamsnap.com/teams/features/rosters

[134] ——. (2015) Schedules. [Online]. Available: https://www.teamsnap.com/teams/features/schedules

[135] ——. (2015) Teamsnap live! [Online]. Available: https://www.teamsnap.com/teams/features/teamsnap-live

[136] ThemeBoy. (2015) Sportspress - manage leagues & sports clubs. [Online]. Available: https://wordpress.org/plugins/sportspress/

[137] ——. (2016) Football club wordpress theme. [Online]. Available: https://www.themeboy.com/football-club/

[138] ——. (2016) Licences & pricing. [Online]. Available: https://www.themeboy.com/sportspress-pro/#pricing

[139] ——. (2016) Wordpress themes designed for sports. [Online]. Available: https://www.themeboy.com/themes/

[140] J. Thorson. (2009) Sports club management. [Online]. Available: https://www.drupal.org/project/sport

[141] TYK. (2016) Tampere arts-oriented senior secondary school. [Online]. Available: http://tyk.info/eng/

[142] w3techs. (2015) Usage of content management systems for websites. [Online]. Available: http://w3techs.com/technologies/overview/content_management/all/

[143] ——. (2016) Usage statistics and market share of wordpress for websites. [Online]. Available: http://w3techs.com/technologies/details/cm-wordpress/all/all

[144] R. Wang. (2015) Avoiding cloud vendor lock-in. [Online]. Available: https://www.enterpriseirregulars.com/102928/avoiding-cloud-vendor-lock/

[145] C. Website. (2015) About us. [Online]. Available: http://www.clubwebsite.co.uk/community/aboutus

[146] ——. (2015) Terms of service. [Online]. Available: http://www.clubwebsite.co.uk/community/tos

[147] Website4sport. (2015) Features. [Online]. Available: http://www.website4sport.com/features

[148] ——. (2015) Pricing. [Online]. Available: http://www.website4sport.com/pricing

[149] Wikipedian. (2015) List of content management systems. [Online]. Available: https://en.wikipedia.org/wiki/List_of_content_management_systems

[150] D. Wlodarz. (2014) Comparing cloud vs on-premise? six hidden costs people always forget about. [Online]. Available: http://betanews.com/2013/11/04/comparing-cloud-vs-on-premise-six-hidden-costs-people-always-forget-about/

[151] Wordpress. (2016) About wordpress. [Online]. Available: https://wordpress.org/about/

[152] wordpress.com. (2015) Wordpress.com. [Online]. Available: https://wordpress.com/

[153] R. Yokokawa. (2015) Node cms comparison 2015. [Online]. Available: http://y-designs.com/blog/node-cms-comparison-2015/

APPENDIX A. BASIC CONTROLLER EXAMPLE

Program 1 *Basic controller example*

```
// Controller module for the match

// Dependencies
const cmUtils = require('../lib/club_manager_utils.js');
const cmMatch = require('../lib/match.js');
const cmMatchStats = require('../lib/match_statistics.js');
const cmTeam = require('../lib/team.js');
const async = require('async');

module.exports = function(pb) {
  // Pencilblue dependencies
  var util = pb.util;

  // Create the controller
  function MatchController(){};

  // Inherits from base controller:
  accessors for template service,
  // localization service, request and response handlers.
  util.inherits(MatchController, pb.BaseController);

  // Subnavigation key
  var SUB_NAV_KEY = 'match_index';

  ////////////////////////////////////////////////////////
  // Render match view
  // @cb = callback(result)
  ////////////////////////////////////////////////////////
  MatchController.prototype.render= function(cb) {
    var self = this;
    var cos = new pb.CustomObjectService();
```

```
    async.waterfall([
      function(waterfallCb) {
        cmMatch.loadByName(self.query.name, cos, util,
                           waterfallCb);
      },
      function(matches, waterfallCb) {
        cmMatchStats.loadByMatch(
                           matches[0]._id,
                           new pb.DAO(),
                           util,
                           function(err, stats) {
          var match = matches[0];
          match.stats = stats;
          waterfallCb(err, match);
        });
      },
      function(match, waterfallCb) {
        cmTeam.getById(
                           match.season.team,
                           cos,
                           util,
                           function(error, team) {
          const results = {
            match: match,
            team: team.name
          };

          waterfallCb(error, results);
        });
      }
    ], function(error, results) {
      // Register angular objects for match controller
      var ngObjects = pb.ClientJs.getAngularObjects(results);
      self.ts.registerLocal(
                'angular_objects',
                new pb.TemplateValue(ngObjects, false)
```

```
        );

    cmUtils.defaultTemplateValues(pb, self,
            function(err) {
        self.ts.load('matchStandAlone',
                function(err, result) {
          if(util.isError(err)) {
            throw err;
          }

          cb({content: result});
        });
      });
  });
};

//////////////////////////////////////////////////////
// Setup up routes for this controller.
// Pencilblue will call getRoutes() for each
// controller in the controllers during initialization
// to register handlers for the defined routes.
//////////////////////////////////////////////////////
MatchController.getRoutes = function(cb) {
  var routes = [
    {
      method: 'get',
      path: '/club-manager/match/',
      auth_required: false,
      content_type: 'text/html',
      // handler is not defined, defaults to render()
    }
  ];

  cb(null, routes);
};
```

```
  return MatchController;
};
```

APPENDIX B. API CONTROLLER EXAMPLE

Program 2 *API controller example*

```
// Inherit from pb base API controller

const matchStats = require('../../lib/match_statistics.js');
const Ajv = require('ajv');
const fs = require('fs');

module.exports = function(pb) {
  // Pencilblue dependencies
  const util = pb.util;

  // Create controller
  function MatchApiController(){}

  // Inherit from pencilblue bas API controller
  util.inherits(MatchApiController, pb.BaseApiController);

  ////////////////////////////////////////////////////////
  // Register controller routes
  // Pencilblue will call getRoutes() for each
  // controller in the controllers folder during
  // initialization to regiser handlers for the routes.
  ////////////////////////////////////////////////////////
  MatchApiController.getRoutes = function(cb) {
    const routes = [
    {
      method: "post",
      path: "/club-manager/api/stats",
      auth_required: true,
      content_type: 'application/json',
      request_body: ['application/json',
                'application/x-www-form-urlencoded',
                'multipart/form-data'
```

```
        ],
      handler: 'saveStats'
    }, {
      method: "post",
      path: "/club-manager/api/deleteStats",
      auth_required: true,
      content_type: 'application/json',
      request_body: ['application/json',
                'application/x-www-from-urlencoded',
                'multipart/form-data'
        ],
      handler: 'deleteStats'
    }, {
      path: "/club-manager/api/getstats/",
      auth_required: false,
      content_type: 'application/json',
      handler: 'getStats'
    }
    ];

  cb(null, routes);
};

////////////////////////////////////////////////////////
//
// Save new statistics
//
////////////////////////////////////////////////////////
MatchApiController.prototype.saveStats = function(cb) {
  const data = this.body;
  const ajv = new Ajv();

  // Get schema
  fs.readFile(
                'plugins/club-manager/schemas/saveStats.json',
                function(err, schemaBuffer) {
```

```
    const schema = JSON.parse(schemaBuffer);
    const valid = ajv.validate(schema, data);
    if(!valid) {
      cb({
        code: 400,
        content: ajv.errors
      });
    }
    else {
      matchStats.save(data, new pb.DAO(), util,
                function(err, result) {
        // Set response content: Send to the client.
        const response = {
          content: result._id
        };

        cb(response);
      });
    }
  });
};

////////////////////////////////////////////////////////
//
// Delete statistics
//
////////////////////////////////////////////////////////
MatchApiController.prototype.deleteStats = function(cb) {
  matchStats.delete(this.body.id, this.body.type,
              new pb.DAO(), util, function(err, result) {
    cb({});
  });
};

////////////////////////////////////////////////////////
//
```

```
  // Get statistics for the given player.
  //
  ////////////////////////////////////////////////////////////
  MatchApiController.prototype.getStats = function(cb) {
    matchStats.loadPlayerTotals(this.query.player,
          new pb.DAO(), util, function(err, result) {
      // Set response content: Send to the client.
      const response = {
        content: result
      };

      cb(response);
    });
  };

  return MatchApiController;
}
```

APPENDIX C. TEMPLATE EXAMPLE

Program 3 *Template example*

```
^tmp_head^
^angular^
<div class="container" ng-controller="TeamController">
  <div class="row">

    <div class="col-sm-3">
      <div>
        ^tmp_logo^
      </div>
      <ul class="nav nav-pills nav-stacked">
        <li>
          <a ng-href="/club-manager/players?team={{team}}">
            Pelaajat
          </a>
        </li>
        <li class="dropdown">
          <a href="#" class="dropdown-toggle"
            data-toggle="dropdown" role="button"
            aria-haspopup="true" aria-expanded="false"
          >
            Kaudet <span class="caret"></span>
          </a>
          <ul class="dropdown-menu">
            <li ng-repeat="season in seasons">
  <a ng-href="/club-manager/season/?name={{season.name}}">
        {{season.name}}
  </a>
            </li>
          </ul>
        </li>
      </ul>
    </div>
```

```
    <div class="col-sm-9">
      <div class="row">
        <div class="col-sm-12">
          <div ng-repeat="post in posts">
            <!-- Aricles -->
            <div ng-if="isArticle(post)">
              ^tmp_elements=article^
            </div>
            <!-- Match reports -->
            <div ng-if="!isArticle(post)">
              ^tmp_match^
            </div>
          </div>
        </div>
      </div>
    </div>

  </div>
</div>
^tmp_angular=team^
^tmp_footer^
```